

# Home

Edit New page

Alan Johnston edited this page 2 weeks ago · [148 revisions](#)

## Welcome to the AMSAT® CubeSatSim Project Wiki, the CubeSat Simulator

The CubeSatSim is a low cost satellite emulator that runs on solar panels and batteries, transmits UHF radio telemetry, has a 3D printed frame, and can be extended by additional sensors and modules. This project is sponsored by the not-for-profit [Radio Amateur Satellite Corporation, AMSAT®](#).

This page is for the new v2.0 hardware (blue boards, v2.0 and later).

- ▼ Pages 147
- Find a page...
- ▼ Home
  - Welcome to the AMSAT® CubeSatSim Project Wiki, the CubeSat Simulator
  - ▶ [1. Main Board 1](#)
  - ▶ [2. Software Install](#)
  - ▶ [3. Ground Station](#)
  - ▶ [4. Main Board 2](#)
  - ▶ [5. Battery Board](#)
  - ▶ [6. Solar Board](#)
  - ▶ [7. Solar Panels and Frame](#)
  - ▶ [8. Board Stack](#)
  - ▶ [9. Final Testing](#)
  - ▶ [Adding New Sensors](#)
  - ▶ [Command and Control](#)
  - ▶ [Creating the CubeSatSim Raspberr...](#)



▶ [CubeSatSim Lite](#)

▶ [CubeSatSim Loaner User Guide](#)

Show 132 more pages...

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/alanbjohnston>



If you have the v1.2 hardware (white PCBs) or earlier, then you need these wiki pages:

<https://CubeSatSim.org/wiki-v1>

If you have the beta v1.3 hardware (blue PCBs), the v2.0 instructions are almost the same, but you can use these wiki pages:

<https://CubeSatSim.org/wiki-beta>

The Bill of Materials (BOM) is here:

<https://cubesatsim.org/bom>

NOTE: If you use the Octopart links in the BOM, the Digikey parts will have the step and identifier information printed on them.



The hardware information including Gerbers is available at:

<https://github.com/alanbjohnston/CubeSatSim/tree/master/hardware/v2.0>

Here you will find documentation about this project and detailed install instructions.

The CubeSatSim has the following features:

- Working solar panels and rechargeable batteries
- Multi-channel voltage, current, and temperature telemetry transmitted in the Amateur Radio UHF band
- Telemetry decoding using [FoxTelem software](#) or APRS software
- Payload microcontroller Raspberry Pi Pico and sensors
- Tape measure monopole, dipole, or SMA antenna
- Integrated Low Pass Filter
- 3D printed frame and solar panels

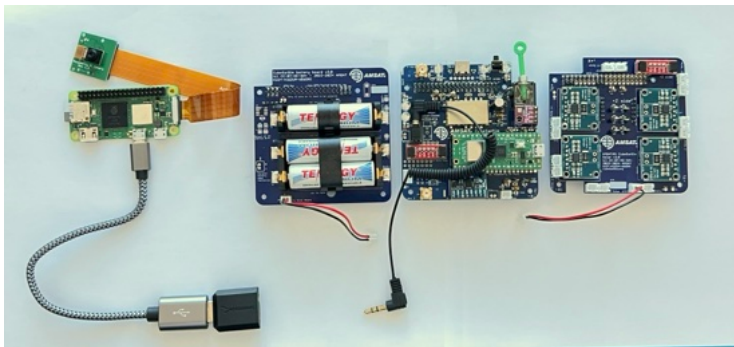
Here are the changes between the v1.2 and the v2.0 hardware and software.

- New FM transceiver module for better frequency stability and simple command and control receiver to change telemetry mode
- More modern and cheaper Raspberry Pi Pico micro controller
- Easily connect additional sensors for the Pico using the Qwiic connector system <https://www.sparkfun.com/qwiic>
- SSTV camera images now display callsign and battery status overlay
- Can be modified to fly as a balloon payload

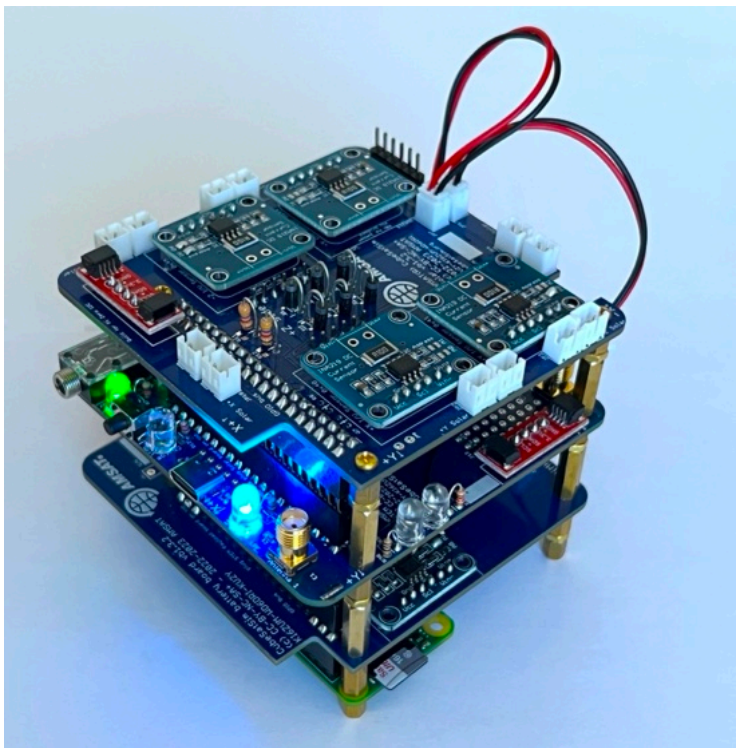
with 500mW FM output for SSTV, APRS, or CW transmissions with software support on Pico for a serial GPS module

- Lower parts cost and easier to source. All parts can now be sourced from electronics distributors and Amazon including easy to find solar panels. New BOM uses Octopart electronic part inventory site with one click distributor ordering
- Redesigned for blue INA219 voltage and current sensors instead of more expensive purple ones
- Battery board now has integrated voltage and current sensor and stronger holder with better performance
- Simpler electrical power system with no boost converter or charge control circuit
- Kits can be built with through hole parts except for a few SMT parts. Fully assembled boards will be available in the future using SMT parts
- Easily connect additional sensors for the Raspberry Pi Zero the Qwiic connector system

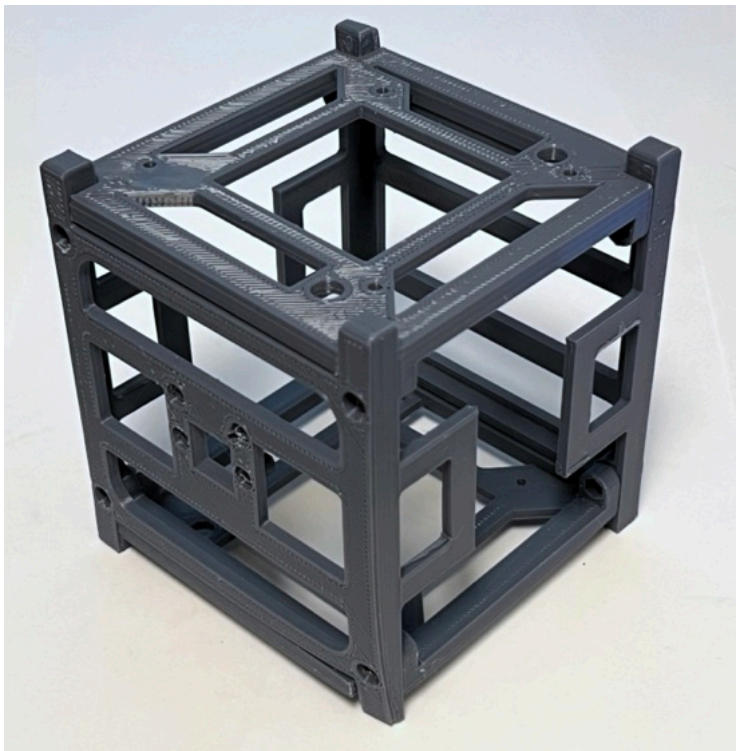
Here are the four boards that make up the complete board stack. Left to right: Raspberry Pi Zero WH with Pi Camera, Battery Board, Main Board with FM module, and Solar Board.



Here is the built board stack:

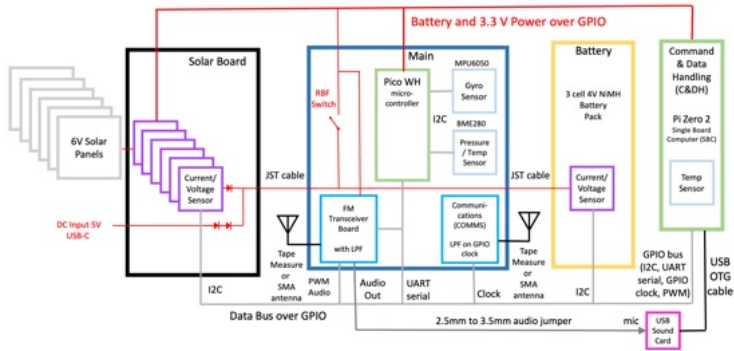


There is a 3D printed frame:



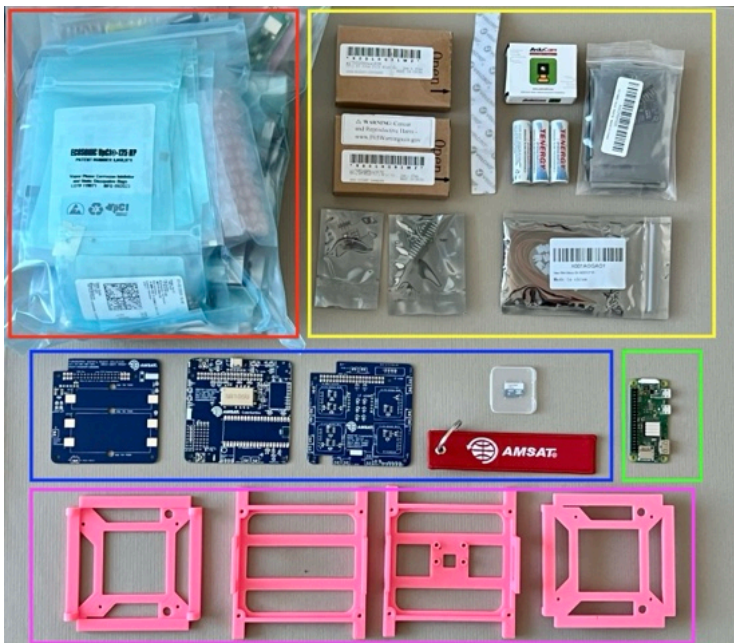
Here is a block diagram of the design:

## CubeSatSim Block Diagram v2.0



Parts List Bill of Materials (BOM) to build the CubeSatSim is available here. Make a copy of this spreadsheet: <https://CubeSatSim.org/bom>

Here is a photo of a kit of parts. Boxed in red are the parts from Digikey, yellow parts from Amazon, blue parts from AMSAT, green parts from various places (use rpilocator.com to find in stock Pi Zero WH), and pink is the 3D parts you print yourself:



Here are the steps to build a CubeSatSim:

- [1. Build the Main Board Part 1](#)
- [2. Install the Software](#)

[3. Build a Ground Station](#)

[4. Continue Building the Main board Part 2](#)

[5. Build the Battery board](#)

[6. Build the Solar board](#)

[7. Assemble the Solar Panels and Frame](#)

[8. Put the Board Stack together and mount in the Frame](#)

[9. Final Testing](#)

+ Add a custom footer



# 1. Main Board 1

[Edit](#) [New page](#)

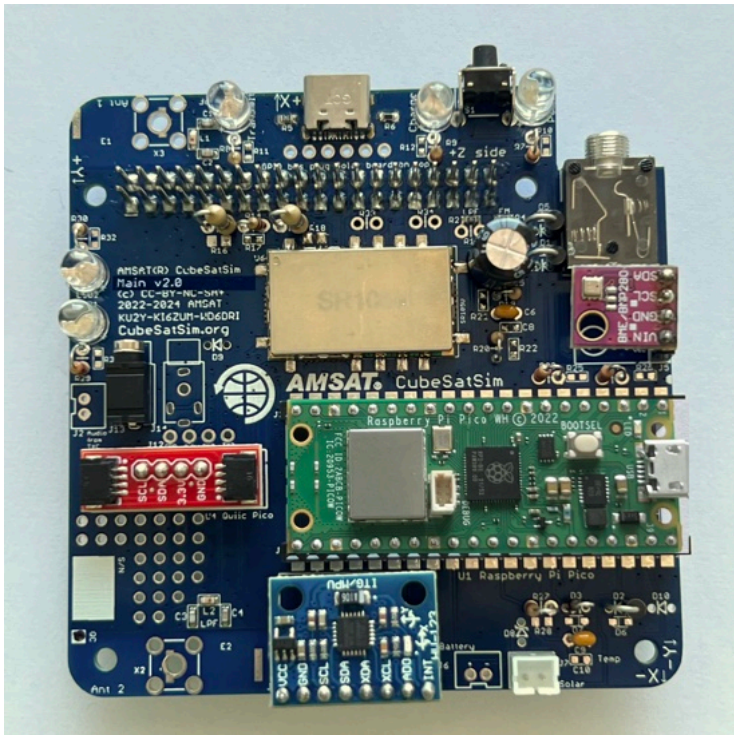
Alan Johnston edited this page 4 days ago · [47 revisions](#)

If the images on this page fail to load, you can [download a PDF of the page here](#).

## 1. CubeSatSim Main Board Assembly Part 1

In these steps, you will start building your Main Board.

These steps are used to build a v2.0 CubeSatSim shown here:



To do this, you will need the parts in the BOM and these tools:

- Pages 147
- Find a page...
- Home
- 1. Main Board 1
  - 1. CubeSatSim Main Board Assembly Part 1
  - Video
  - Checklist
  - Main Board Instructions
  - Schematic
  - Assembly
- 2. Software Install
- 3. Ground Station
- 4. Main Board 2
- 5. Battery Board
- 6. Solar Board
- 7. Solar Panels and Frame
- 8. Board Stack
- 9. Final Testing
- Adding New Sensors

- Safety glasses (to protect eyes while soldering or trimming leads)
- Soldering iron and solder (I use lead-free solder, but leaded solder is easier to work with)
- Needle nose pliers (to bend leads and hold parts)
- Side cutters (to trim leads)

Other tools that are helpful:

- [Blue mounting putty](#) (to hold components in place while soldering)

## Video

---

Here is a [video showing this assembly step](#). Note this video shows the v1.3.2 boards - the assembly is the same except for the addition of diode D4.

## Checklist

---

Here is the Bill of Materials (BOM): <https://CubeSatSim.org/bom> This board is the first set of parts labeled "Main Board"

The BOM has a sheet "By Steps" which lists the parts needed for each step in order. If you have a Google account, you can make a copy of this spreadsheet ("File" then "Make a Copy") and check off each part as you install it.

For example, here is the checklist for this step:

- ▶ [Command and Control](#)
- ▶ [Creating the CubeSatSim Raspberr...](#)
- ▶ [CubeSatSim Lite](#)
- ▶ [CubeSatSim Loaner User Guide](#)














Show 132 more pages...

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/alanbjohnston>



<input checked="" type="checkbox"/>	Ref	Item	Qty	Location	Image
<input type="checkbox"/>		Alt 4a: Main Board PCB with SMT components	1		
<input type="checkbox"/>	J1	GPIO 20x2 female stacking header extra long	1	Bottom	
<input type="checkbox"/>	S1	Pushbutton switch, SPST RA-SPST	1	Top	
<input type="checkbox"/>	X1	SC1464-ND 3.50mm Headphone Phone Jack	1	Top	
<input type="checkbox"/>	R7	1k Ohm resistor	1	Top	
<input type="checkbox"/>	LED3	Green LED	1	Top	
<input type="checkbox"/>	R9	220 Ohm resistor	1	Top	
<input type="checkbox"/>	LED5	Red LED	1	Top	
<input type="checkbox"/>	R8	100 Ohm resistor	1	Top	
<input type="checkbox"/>	LED4	Blue LED	1	Top	
<input type="checkbox"/>	D1, D4	IN5817 diode	2	Top	
<input type="checkbox"/>	R13, R	68 Ohm resistor, 1/2 W	2	Top	
<input type="checkbox"/>	R14	180 Ohm resistor	1	Top	

## Main Board Instructions

If you don't have the ability to do SMT soldering, you can instead use an external Band-Pass Filter with SMA connectors. For example, this BPF works quite well

<https://www.amazon.com/gp/product/B07R8Y1P>

[M4/](#). For the USB-C connector J9, you can use J8, the Sparkfun USB-C breakout

<https://www.sparkfun.com/products/15100>. And

instead of the 1k SMD resistor R2, you can install a through hole 1k resistor in R1.

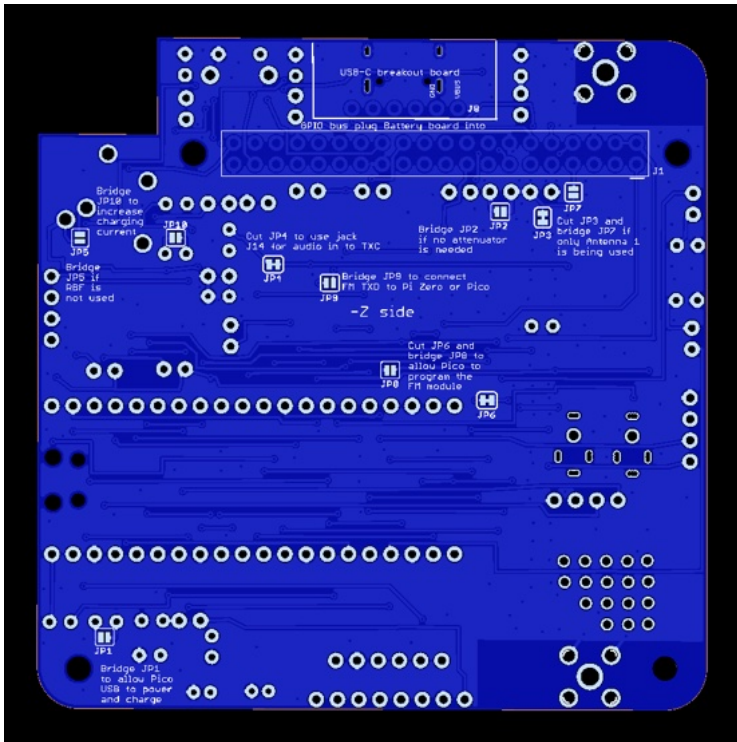
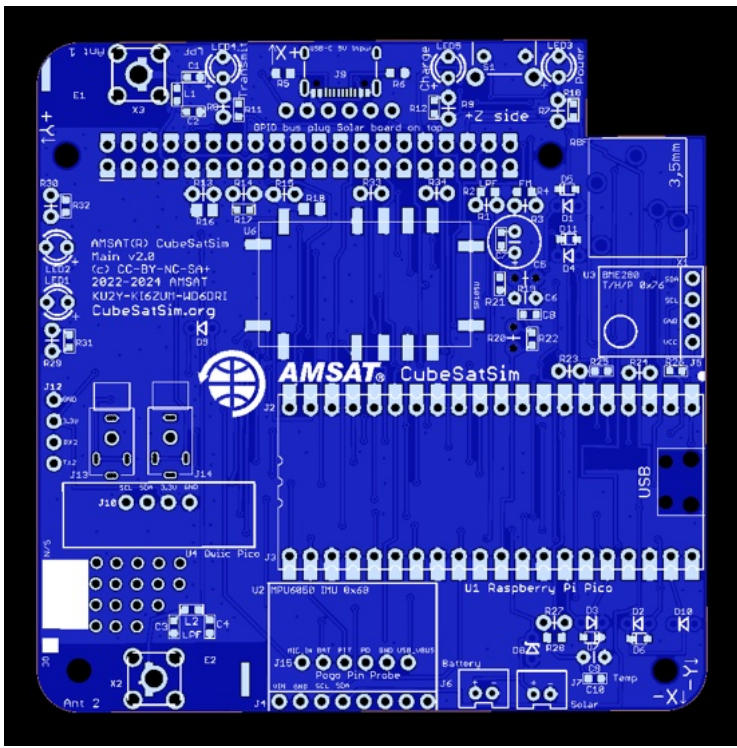
Here is the PCB with the SMT components installed:





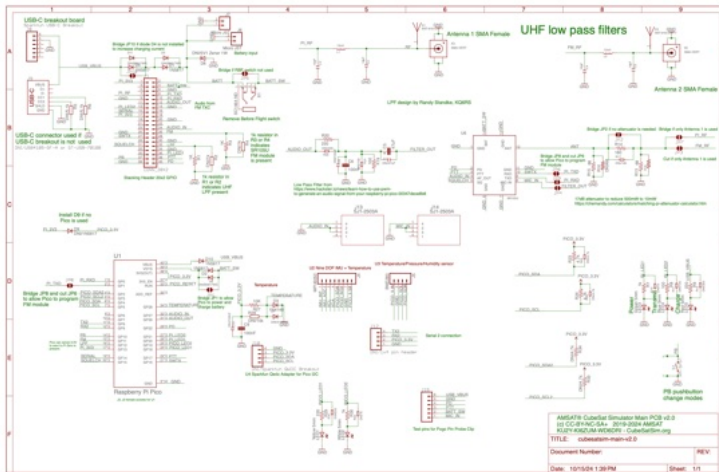
If you aren't good with SMT soldering, you can make a board with just the Low Pass Filters and the FM Transceiver module as SMT, with the USB-C and other components as through-hole soldering. Here are the steps to [install the SMT components](#). (The SMT parts are in the BOM in the SMT Parts tab).

Here's the v2.0 PCB:



# Schematic

Here's the schematic for the v2.0 Main board for reference:

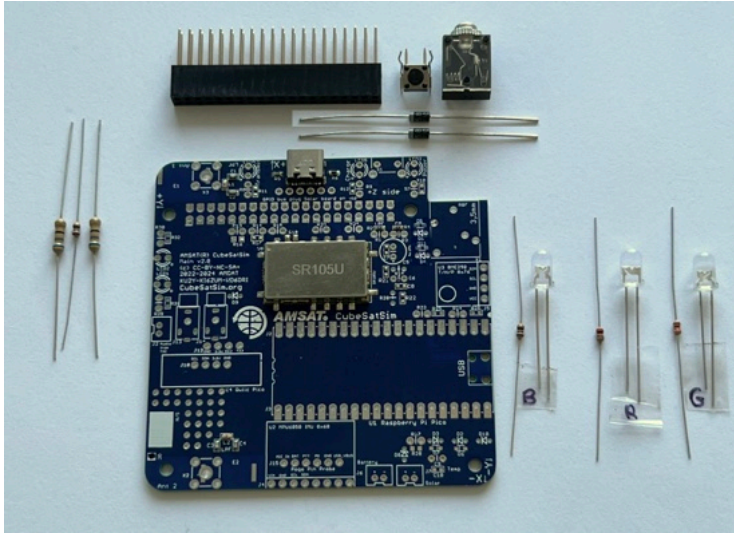


[https://github.com/alanbjohnston/CubeSatSim/blob/42b777577a17e1c3bdc2f7bfbf3d5b9bec08124d/hardware/v2.0/cubesatsim-main-v2.0\\_schematic.pdf](https://github.com/alanbjohnston/CubeSatSim/blob/42b777577a17e1c3bdc2f7bfbf3d5b9bec08124d/hardware/v2.0/cubesatsim-main-v2.0_schematic.pdf)

## Assembly

The printed outlines on the board indicate that a part should be installed on that side, with the pins soldered on the other side. For components with multiple pins, solder just one pin first, then flip the board over and verify that it is flush with the board and straight. It is a lot easier to fix this with only one pin soldered than with multiple pins. It is also one last check to make sure you are mounting it on the right side. For some parts, such as resistors or diodes, mounting on the wrong side won't matter. For others, such as LEDs and switches and connectors, mounting on the wrong side will make it more difficult to mount the solar panels. But other parts, such as the stacking GPIO headers or the INA219 boards, must be mounted on the correct side, so double check carefully.

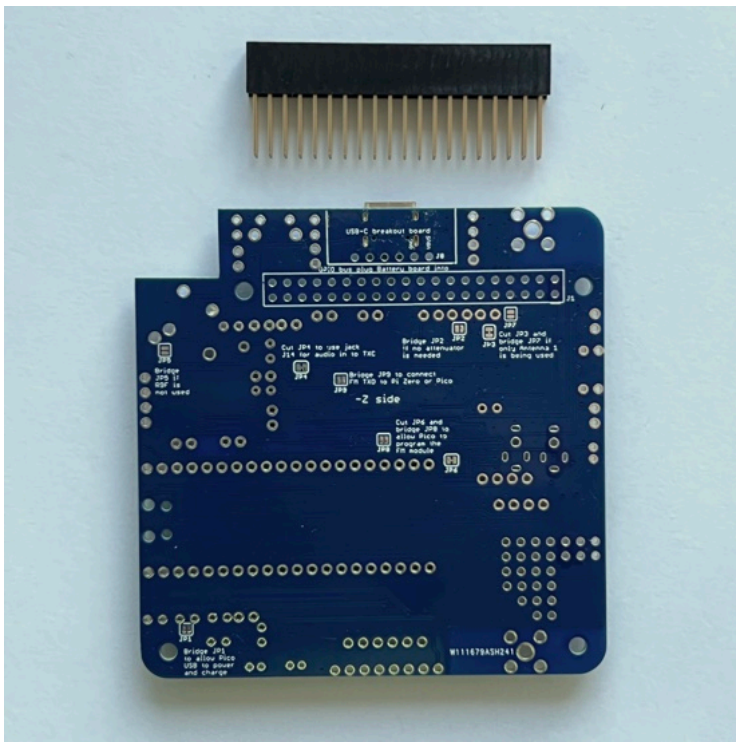
This photo shows the set of parts to be mounted in the next steps on the PCB:



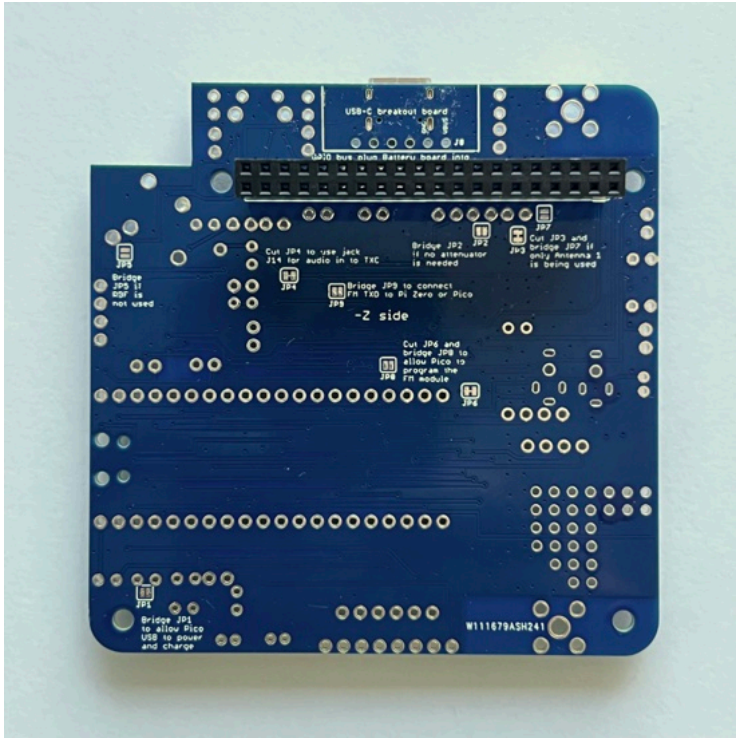
Start with the stacking GPIO header J1. We use the Raspberry Pi GPIO as our spacecraft bus, and it connects the Pi and the three boards together. We will insert the stacking GPIO header on the bottom of the PCB and solder the pins on the top of the PCB.

**IMPORTANT: Do not trim the leads on this stacking header, since the long leads are needed to plug the next board on top!** It is also important that you don't get excess solder on the top of the GPIO pins or it might be hard to plug then next board in.



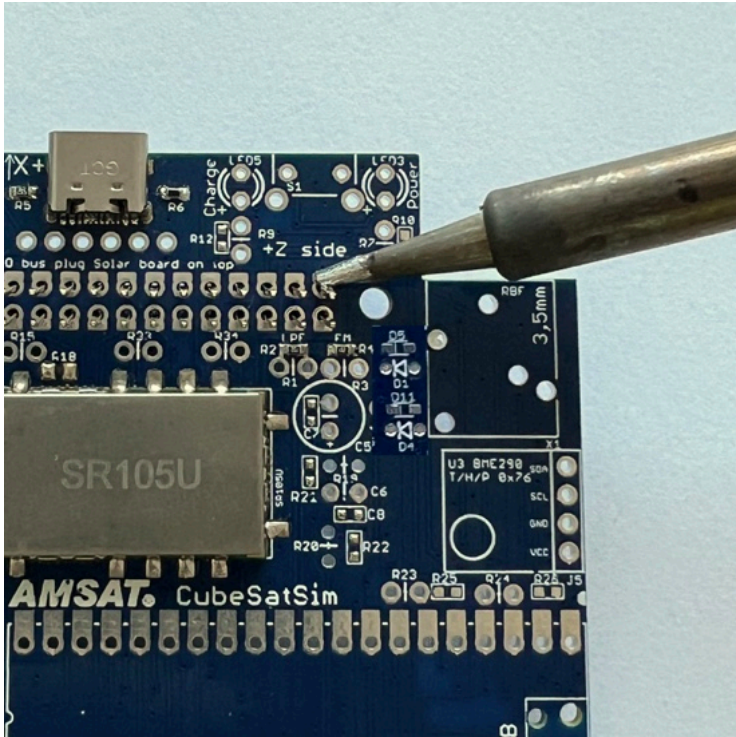


Flip the PCB upside down and insert the GPIO header:

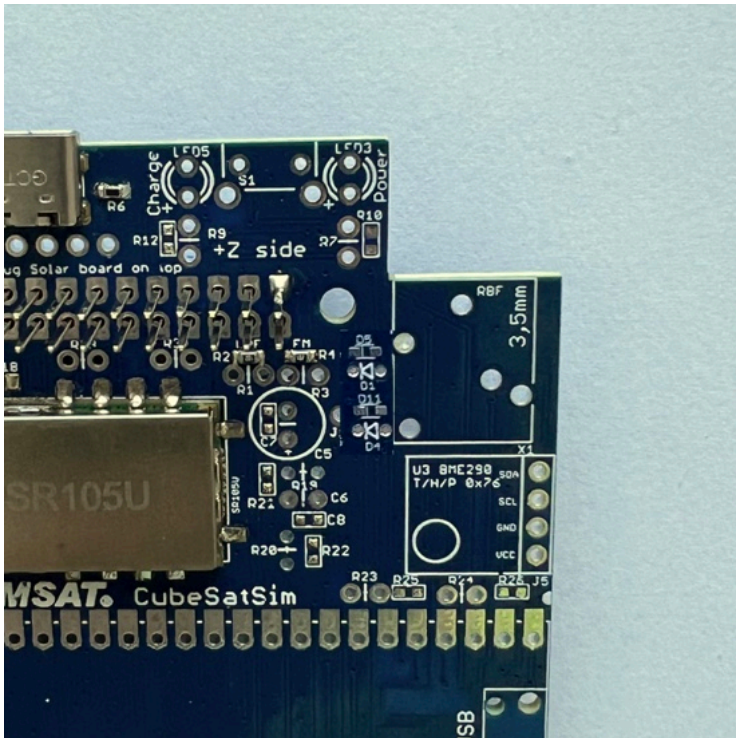


Flip the PCB to the top to get ready for soldering. To make it easier solder, the PCB has enlarged pads making it easier to apply heat.

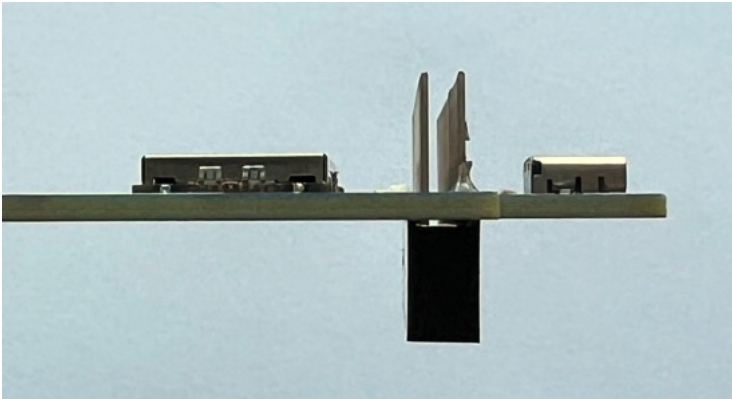
If you place your soldering iron on this pad touching the pin, you should be able to get heat to both the pin and pad to solder it.



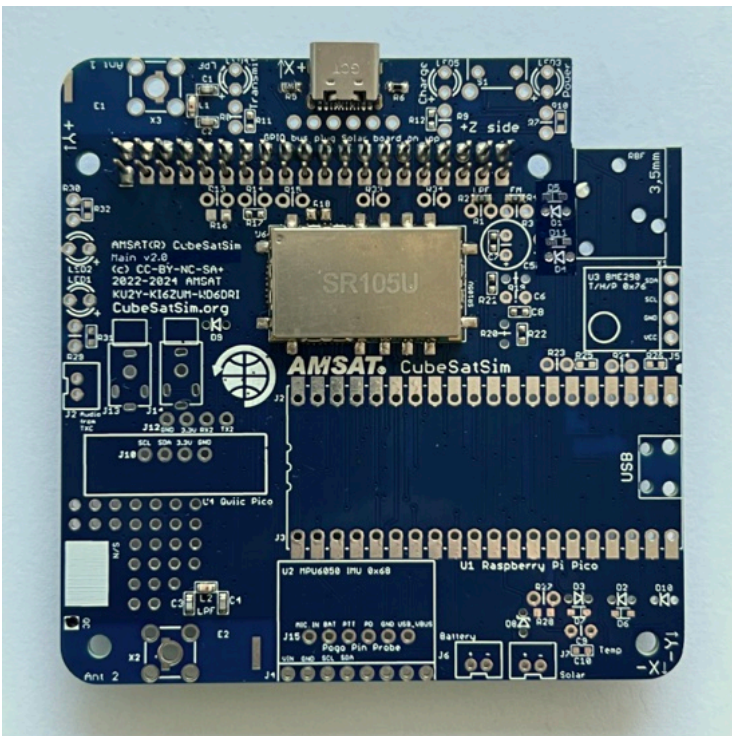
Solder only one pin on the corner of the connector:



Double check that it is pressed down all the way and straight before going back and soldering any other pins:

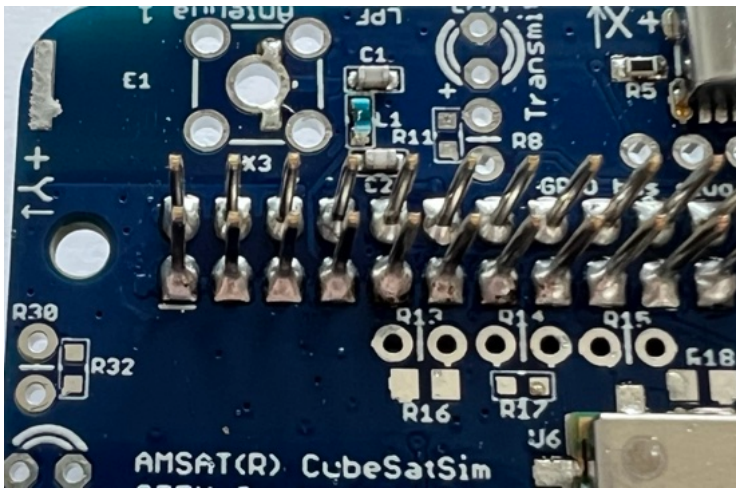


If everything is good, solder one pin on the other corner and check to make sure it is straight and flush with the PCB. Check the connector again carefully. If it looks good, go ahead and solder one row of pins. Here's how it looks on the top of the PCB with one row soldered in:



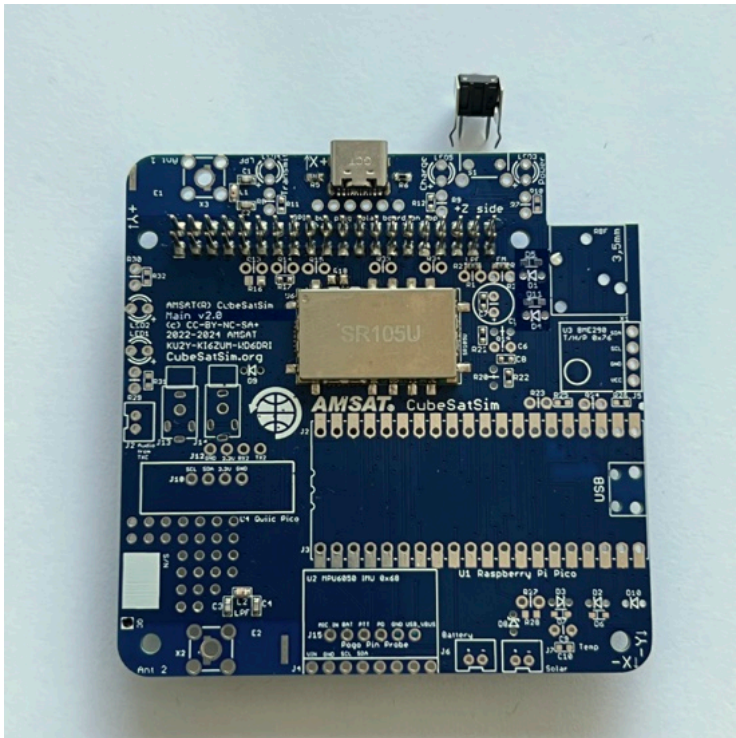
Then solder the 2nd row. The solder should fill the pad but not go up the pin or it will be difficult to insert another board into the connector.





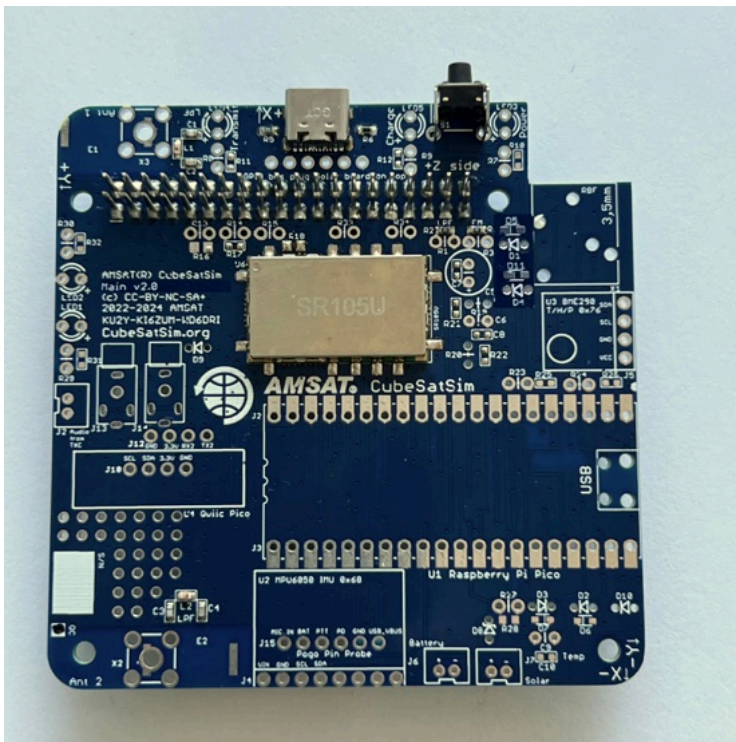
The rest of the components are inserted in the top of the PCB and soldered on the bottom of the PCB.

Next is the push button switch S1. This momentary push button switch is used to change modes or shutdown when it is running:

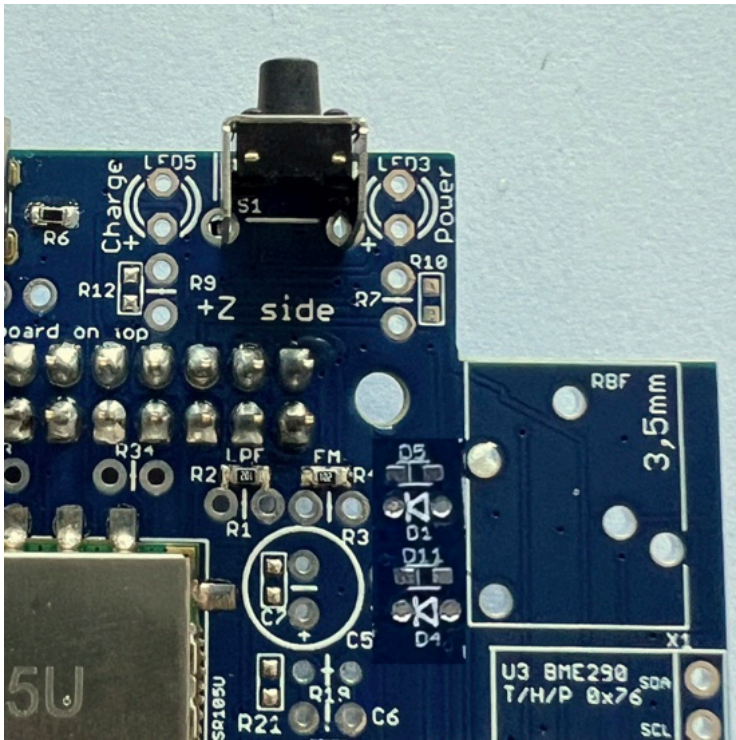


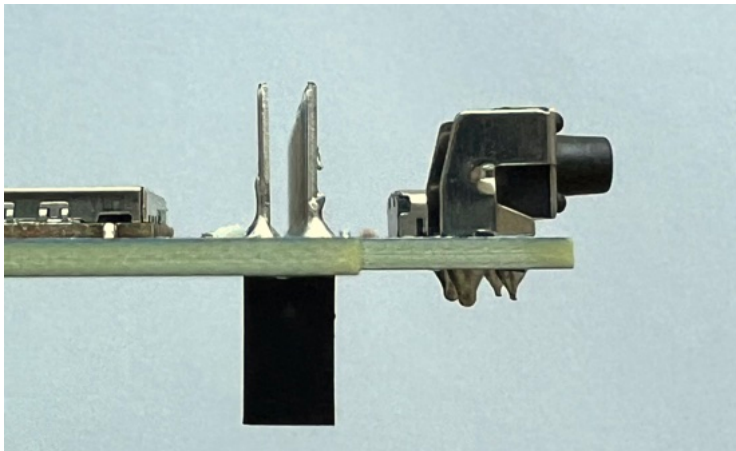
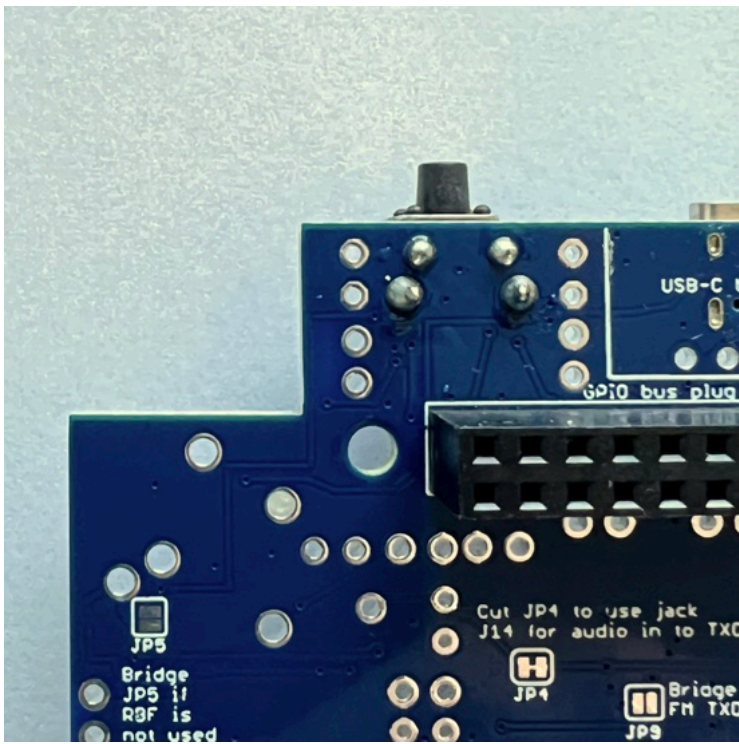
Insert the switch in the top and solder the four leads on the bottom of the PCB:





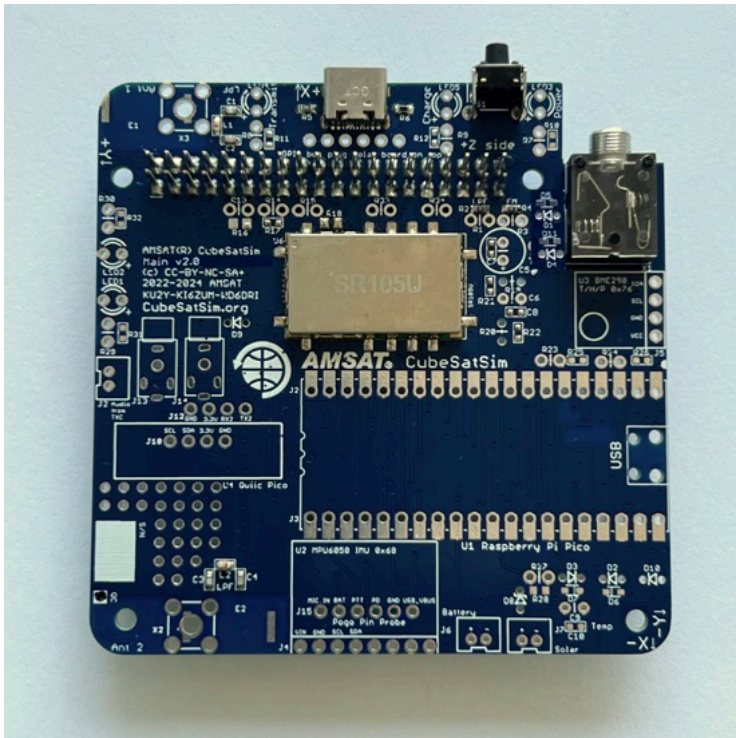
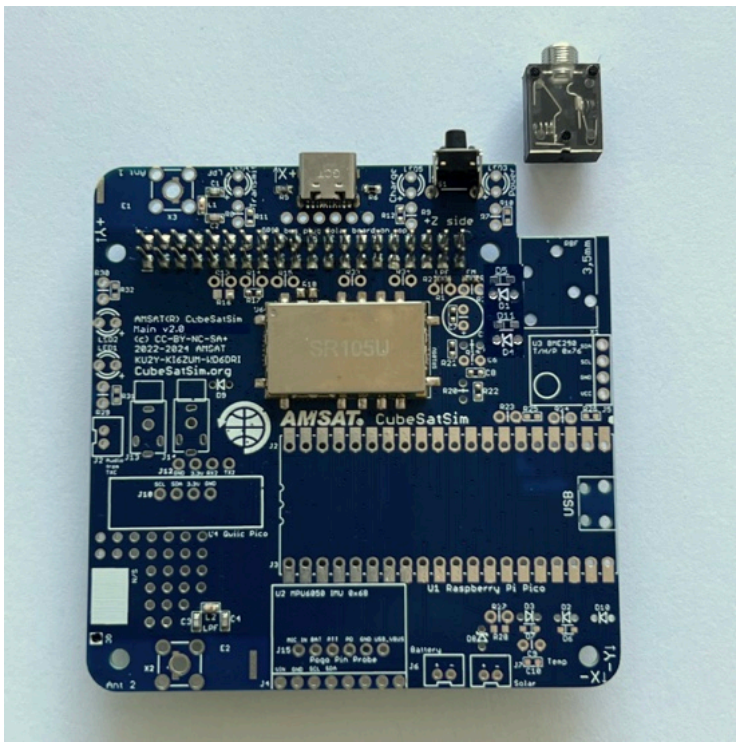
Here's how it looks up close:



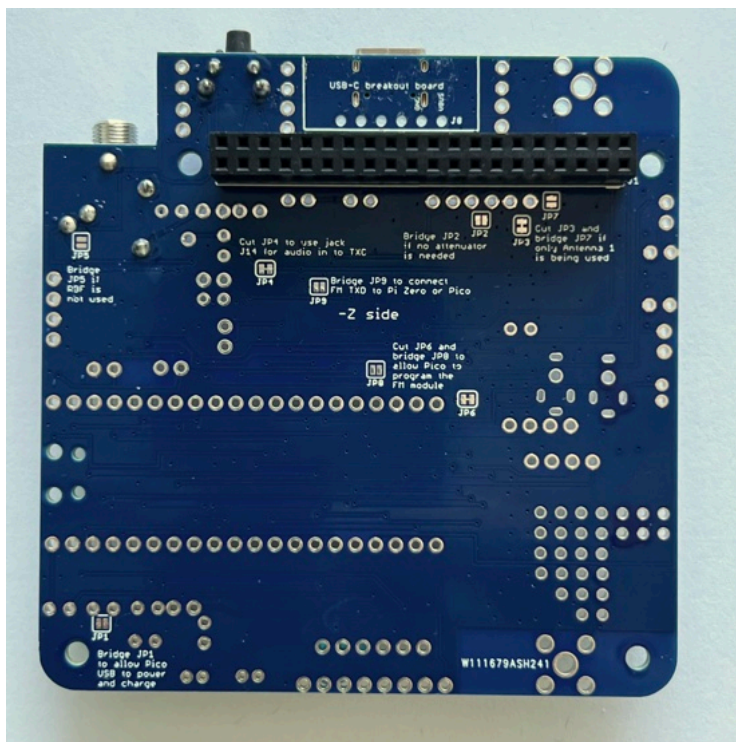


Then, solder X1 the 3.5mm audio jack, which has a contact switch (you can see the contacts with the springs if you look through the clear cover) for our Remove Before Flight (RBF) switch. This switch disconnects power from the CubeSatSim for transporting or storage, or, on a real CubeSat, prior to launch.



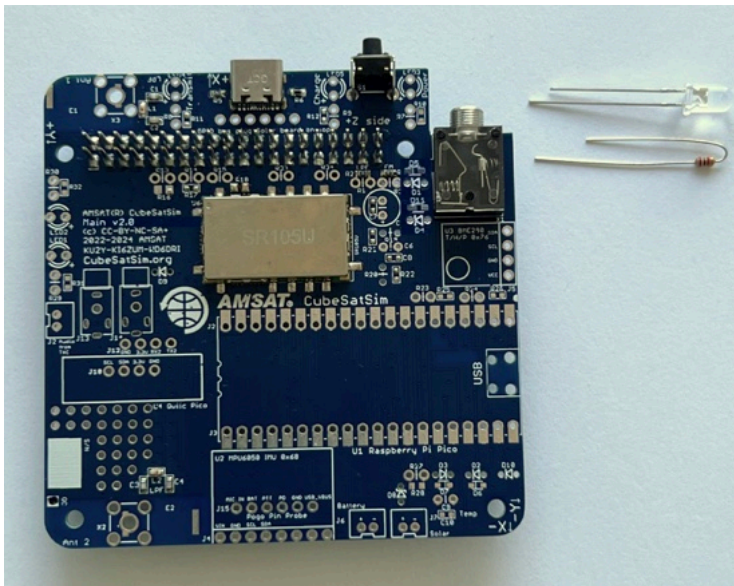


Flip the PCB over and solder the 5 leads:

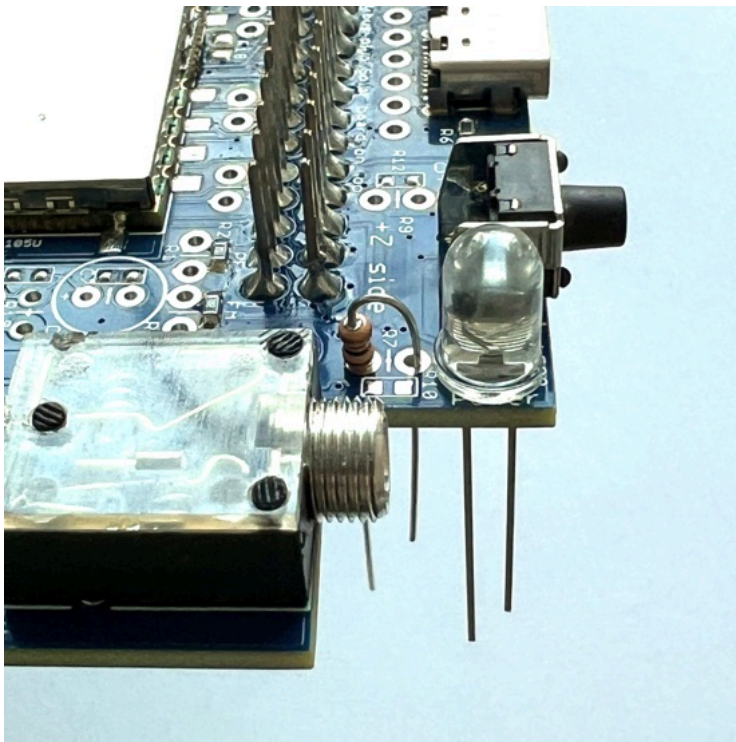


Next we will solder the resistors R7, R8, and R9, and the three LEDs LED3, LED4, and LED5. If you are using clear lens LEDs, be careful not to mix up the colors! If you are not sure, test them with 3.3V and resistor R11 to see what color it is. LEDs need to be installed with the correct polarity (one lead is + polarity and the other lead is - polarity) or it will not illuminate. The longer leg on the LED is the '+' lead and should be away from the edge of the PCB. Also, if you look at the LED lens from the top, it is circular but there is a flat side that marks the '-' side. So the flat side of the LED should be towards the edge of the PCB.

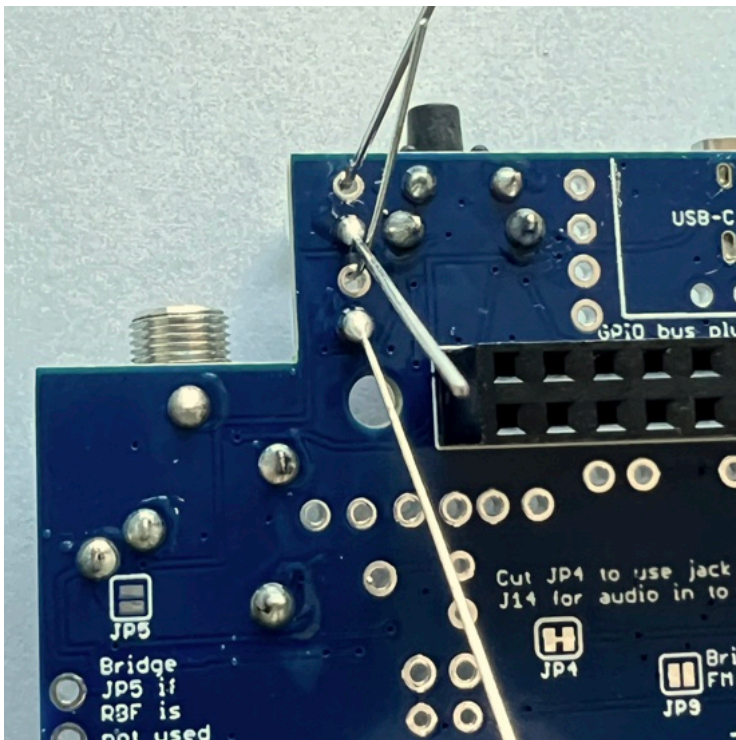
To save space on the PCB, all resistors and diodes are mounted vertically, so you have to bend over one of the leads to get a U-shape, allowing it to be inserted into the two closely-spaced holes. This photo shows the green Power LED3 and 1k Ohm resistor R7 (the color bands on the resistor are brown black red - see [https://en.wikipedia.org/wiki/Electronic\\_color\\_code](https://en.wikipedia.org/wiki/Electronic_color_code) for details) which are about to be inserted next to the RBF switch. Note the longer plus lead of the LED is on the bottom here:



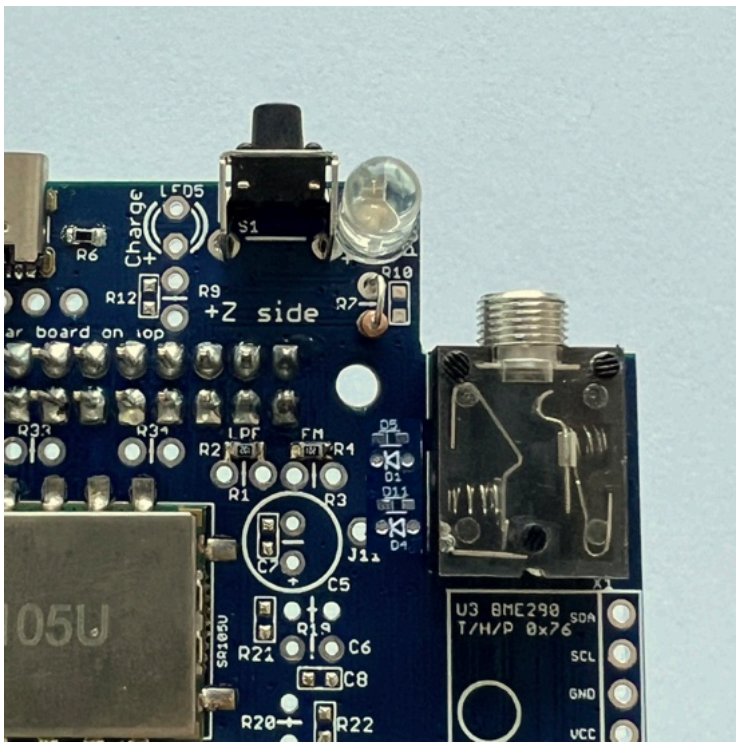




Insert the LED and resistor then flip the PCB upside down. Bend the leads slightly so the part is held in place. Solder one pin on each part:

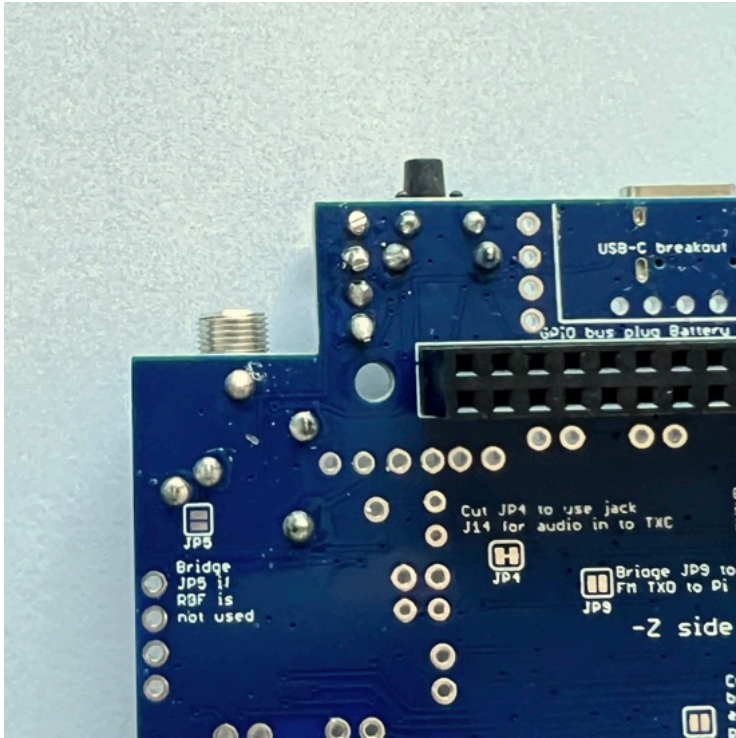


Make sure the parts are straight and inserted all the way into the PCB:



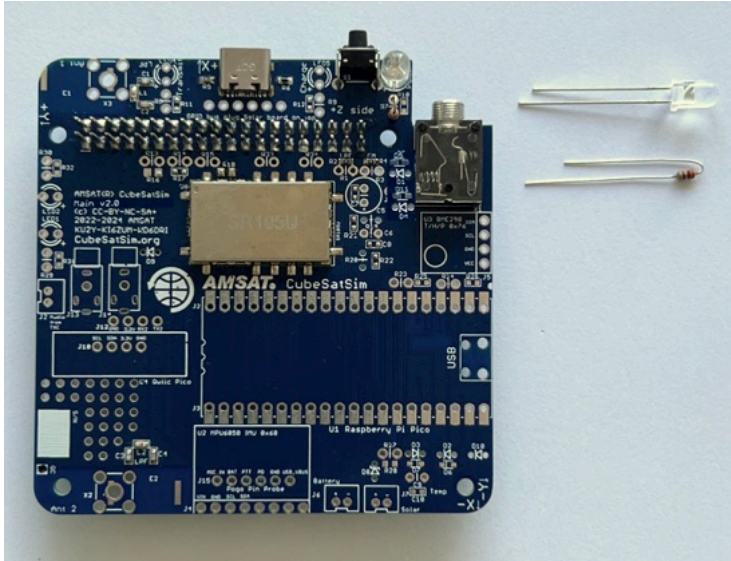
Solder the other pins, then trim the excess leads with side cutters.

Here's how the resistor and LED leads look after they have been soldered on the bottom of the PCB and the excess leads trimmed:

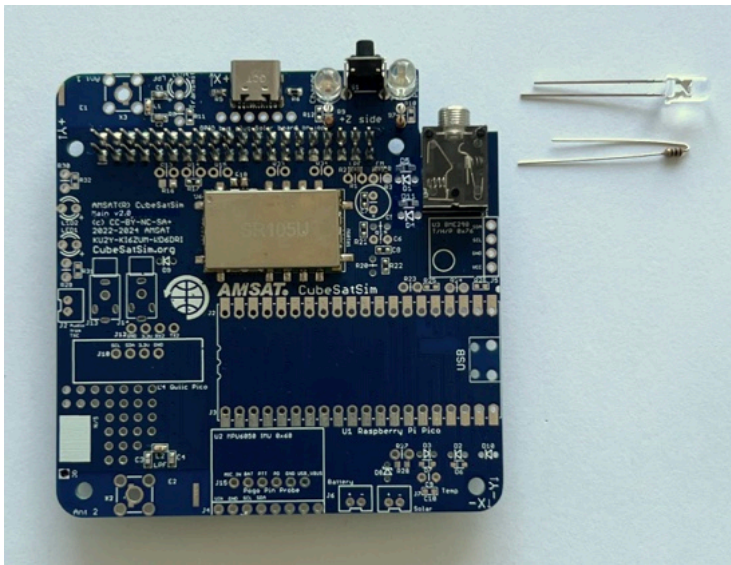




Next, install the red Charge LED5 and 220 Ohm resistor R9 (color bands **red red brown**),

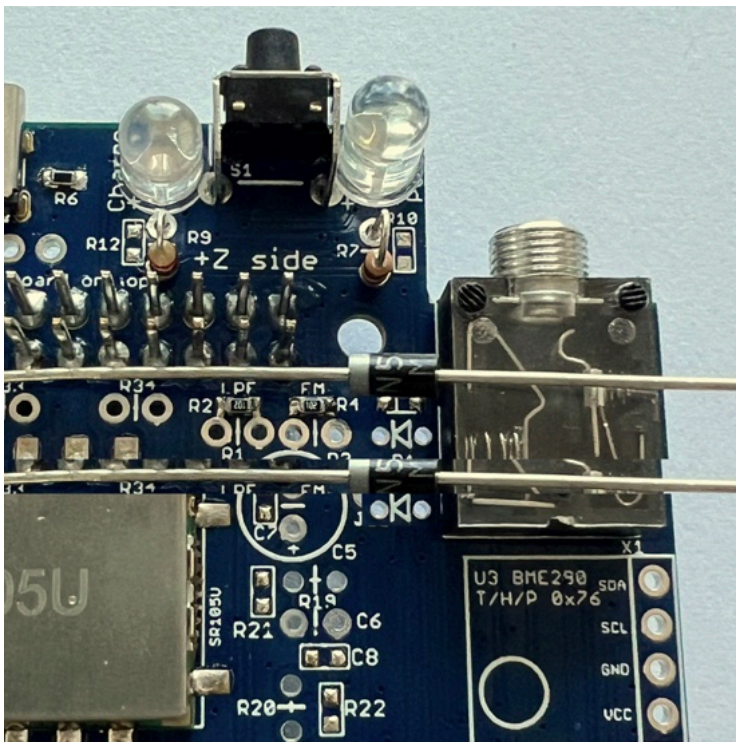


Then install the blue Transmit LED4 and 100 Ohm resistor R8 (color bands **brown black brown**):

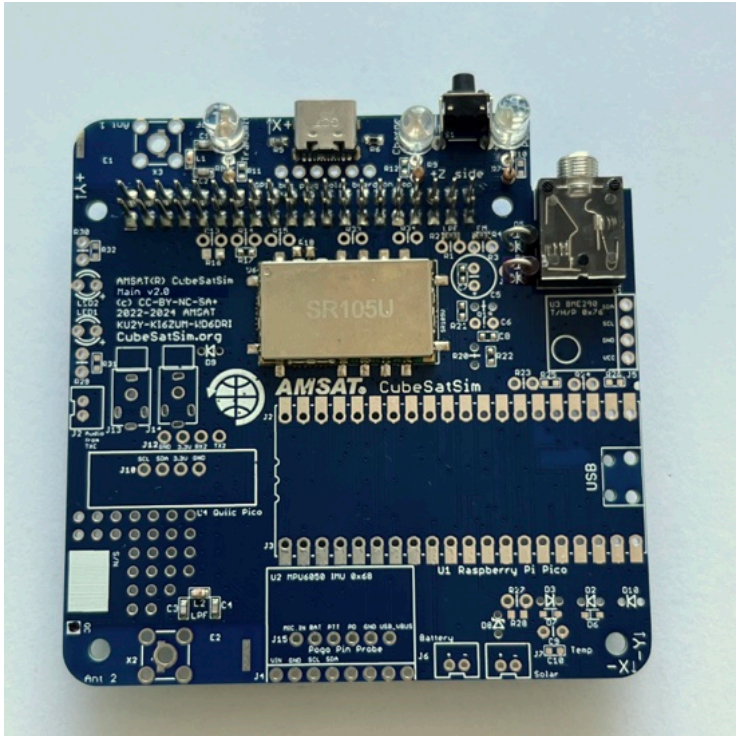


Next install the IN5817 diodes D1 and D4. If you look closely, you can see the numbers 58 and 17 written on the diode. Diodes have polarity marked with a band on one side. In this photo, the white band on the diodes is to the left which matches the symbol markings for D1 and D4:

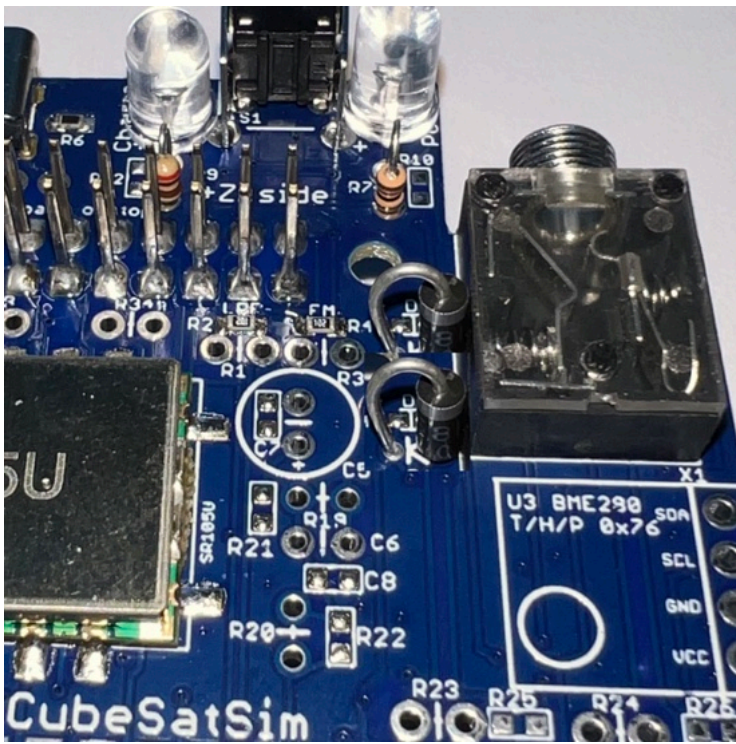




Diodes are mounted vertically like the resistors. Note the band is on the upper side in this photo:

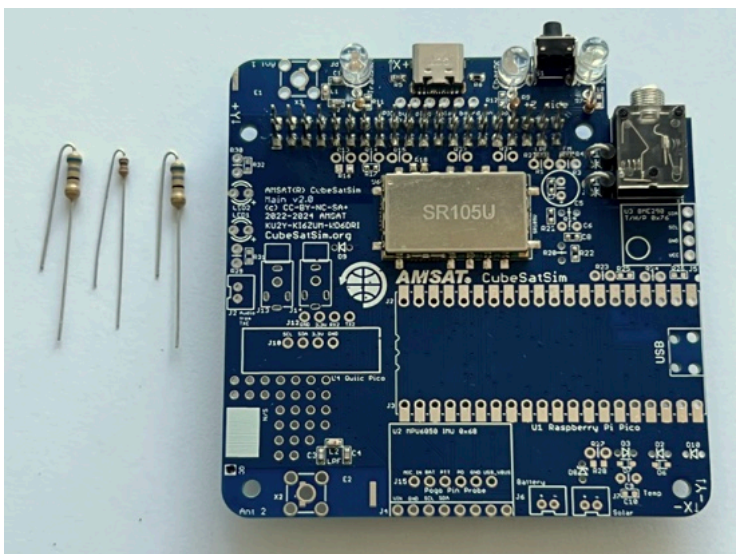


Here's a closeup of the diodes inserted:

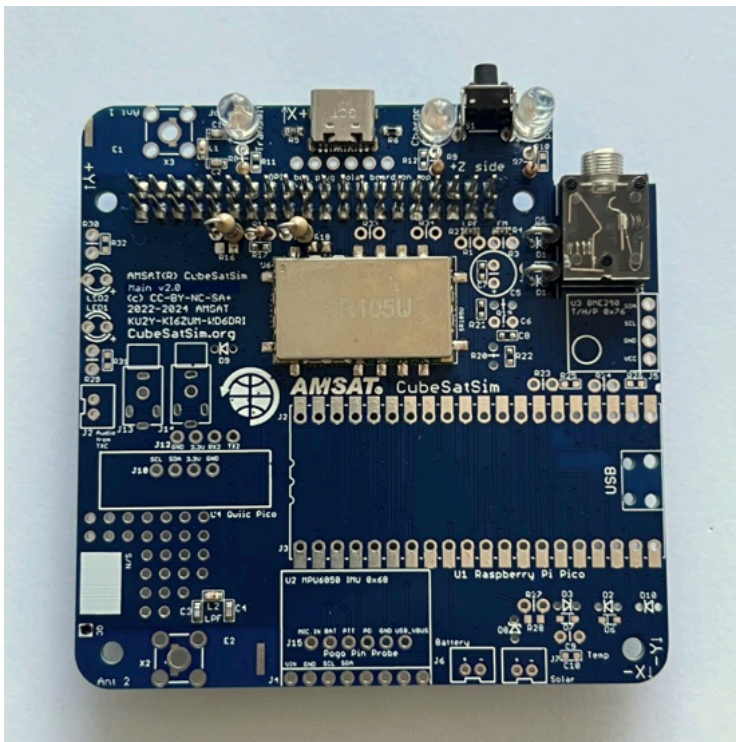


If either diode is backwards, when we plug in the USB-C power cable, the Red LED will illuminate but the board will not power up!

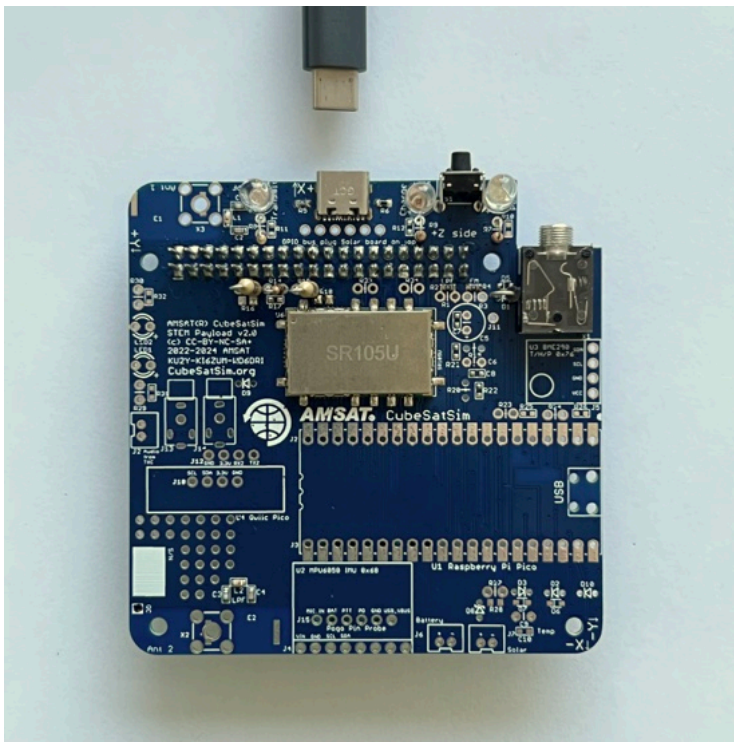
Next install the two 68 Ohm resistors R13 and R15 (big 1/2 Watt resistors with color bands **blue gray black**) and the 180 Ohm resistor R14 (color bands **brown gray brown**) in the attenuator circuit:



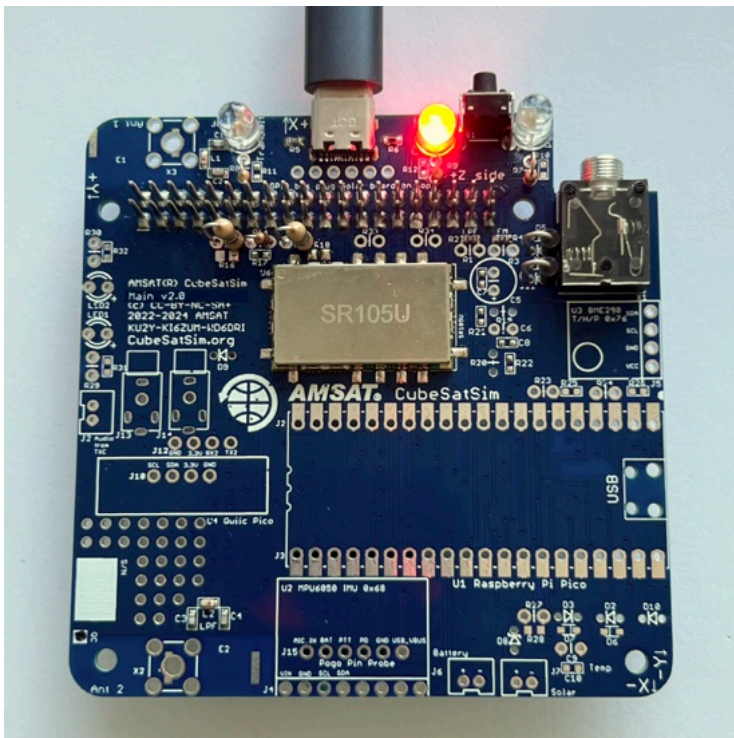




We can now test the Red LED using the USB-C connector.



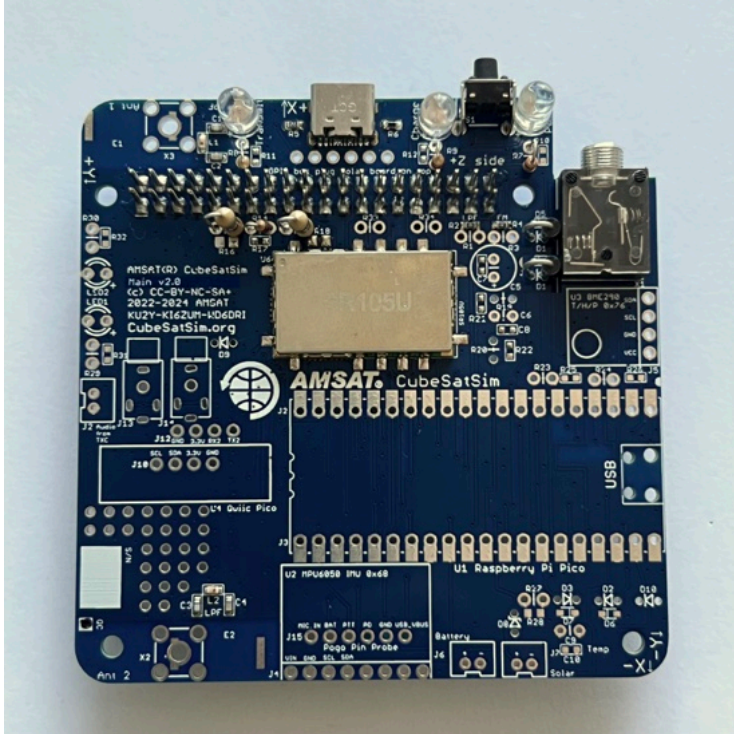
Plug your USB-C cable into a power plug or a computer, then plug it into the USB-C connector on the PCB and the Red LED should illuminate.



If it doesn't, check LED5, resistor R19, and the USB-C connector.

The rest of the LEDs need the Raspberry Pi Zero WH programmed and plugged in, which is our next step!

You have completed Part 1 of building the Main board. Here's how the board looks so far:



Next step is to [Install the Software](#).

+ Add a custom footer



# 2. Software Install

Edit New page

Alan Johnston edited this page 2 weeks ago · [36 revisions](#)

If the images on this page fail to load, you can [download a PDF of this page here..](#)

## 2. Software Install

Software needs to be installed on the Raspberry Pi Zero 2 and on the Raspberry Pi Pico W.

Note: You can run the software on the Pi Zero, but the Pi Zero 2 is recommended due to the CPU load of Command and Control. Note: You can run the Payload software on the Pico, but the Pico W is recommended as future features will make use the WiFi capability.

### Raspberry Pi Zero 2 Software

There are two ways to get the CubeSatSim software for your Pi.

One option is to download the disk image file and write it to a 16GB or larger micro SD card. The image is based on Raspberry Pi OS (Rasbian) Lite, Bullseye (Legacy) version dated December 2023.

- ▼ Pages 147
- Find a page...
- ▶ [Home](#)
- ▶ [1. Main Board 1](#)
- ▼ [2. Software Install](#)
  - 2. Software Install
    - Raspberry Pi Zero 2 Software
    - Video
    - Checklist
    - Disk Image Option Steps
    - Logging into your Raspberry Pi
      - Wi-Fi login
      - Ethernet over USB
      - Changing the hostname
    - Installation Script Option Steps
    - Raspberry Pi Pico Software
      - Software installation
      - Viewing Software Logs
- ▶ [3. Ground Station](#)
- ▶ [4. Main Board 2](#)
- ▶ [5. Battery Board](#)

All the software is installed, you just need to login to change your password and set your amateur radio callsign if you have one. You can run the `update` script to update all packages and update and compile the latest CubeSatSim software.

The other option is to start with a Bullseye Raspberry Pi OS (Rasbian) image and run the installation script `install` which will install and compile all the related software.

## Video

---

Here is a video showing how to install the software, although it shows the v1.2 software, the steps are the same except you will need to use the v2.0 image as the instructions show below: <https://youtu.be/swWasXBH7PU>

## Checklist

---

The BOM has a sheet "By Steps" which lists the parts needed for each step in order. <http://cubesatsim.org/bom> If you have a Google account, you can make a copy of this spreadsheet ("File" then "Make a Copy") and check off each part as you install it.

For example, here is the checklist for this step:

- ▶ [6. Solar Board](#)
- ▶ [7. Solar Panels and Frame](#)
- ▶ [8. Board Stack](#)
- ▶ [9. Final Testing](#)
- ▶ [Adding New Sensors](#)
- ▶ [Command and Control](#)
- ▶ [Creating the CubeSatSim Raspberr...](#)
- ▶ [CubeSatSim Lite](#)
- ▶ [CubeSatSim Loaner User Guide](#)







Show 132 more pages...

+ Add a custom sidebar

### Clone this wiki locally

<https://github.com/alanbjohnston>



<input type="checkbox"/>	Step 2. Software Install				<a href="https://github.com/alanbjo">https://github.com/alanbjo</a>
<input type="checkbox"/>	<b>Item</b>		<b>Qty</b>	<b>Location</b>	<b>Image</b>
<input type="checkbox"/>	USB-C cable and power plug		1		
<input type="checkbox"/>	Alt 2a: Pi Zero WH (with pre-soldered headers)	<input checked="" type="checkbox"/>	1		
<input type="checkbox"/>	Alt 2b: Pi Zero W	<input type="checkbox"/>	1		
<input type="checkbox"/>	2x20 male header GPIO		1		
<input type="checkbox"/>	16GB micro SD Card with cubesatsim image		1		
<input checked="" type="checkbox"/>	Alt 1a: Pico WH with headers	<input checked="" type="checkbox"/>	1	Top	
<input type="checkbox"/>	Alt 1b: Pico W	<input type="checkbox"/>	1	Top	
<input type="checkbox"/>	Alt 1b: 20 pin male header for Pico		2	Top	

## Disk Image Option Steps

You can write a pre-built Raspberry Pi disk image to your SD card. Your Pi then should boot up immediately and run the CubeSatSim software. You can configure a few settings, but it will run right away.

[Here are the steps we use to create the Raspberry image](#), so you can create your own if you want.

You will need a micro SD card with at least 16GB of capacity and a micro SD card writer and free software such as balenaEtcher:

<https://www.balena.io/etcher/> You will need to have administrator privileges on your computer in order to do this.



The image file is larger than 1GB, but with balenaEtcher you can write it from the URL without having to download it first. When you run balenaEtcher, select the Flash from URL option with this URL:

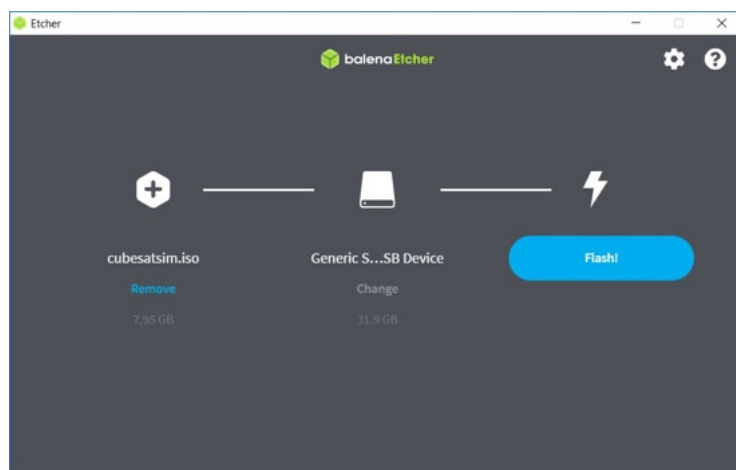
<http://cubesatsim.org/download/cubesatsim-v2.0.iso.zip> Here is the readme file for the

image:

<http://cubesatsim.org/download/cubesatsim-readme.pdf> Insert your micro SD card - the card

will be erased in the process of writing. Click on Select Target and select your SD card writer.

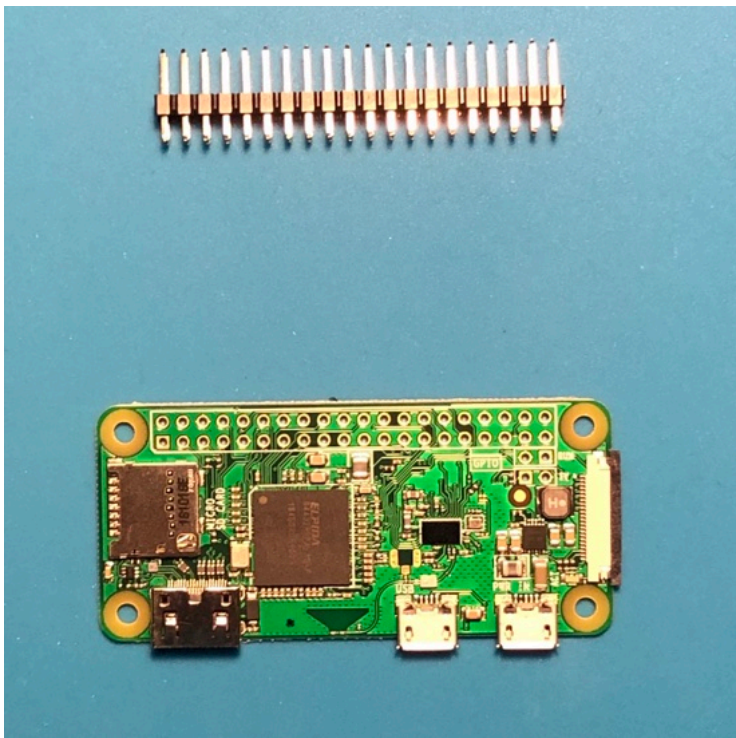
The Size shown here should match the size of your micro SD card.



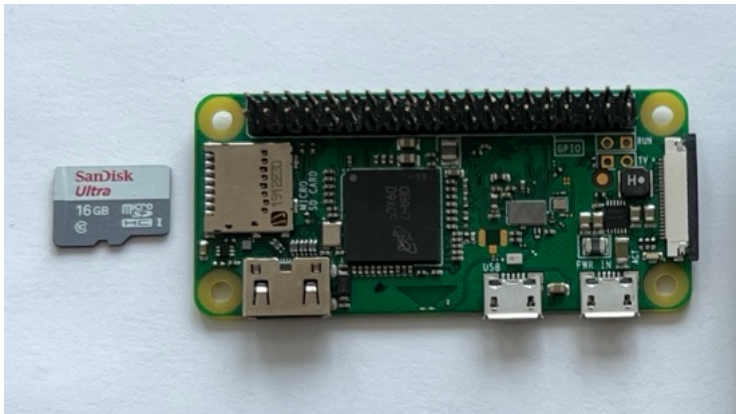
Then select Flash to begin writing. You will need to input your administrator password, then the write will begin. It will take a while.

Alternatively, you can first download the disk image by opening <http://cubesatsim.org/download/cubesatsim-v2.0.iso.zip> in your web browser, then choose the Flash from File option in balenaEtcher then select the image file you downloaded.

If your Raspberry Pi Zero 2 does not have the GPIO header pins installed, you will need to solder them in:

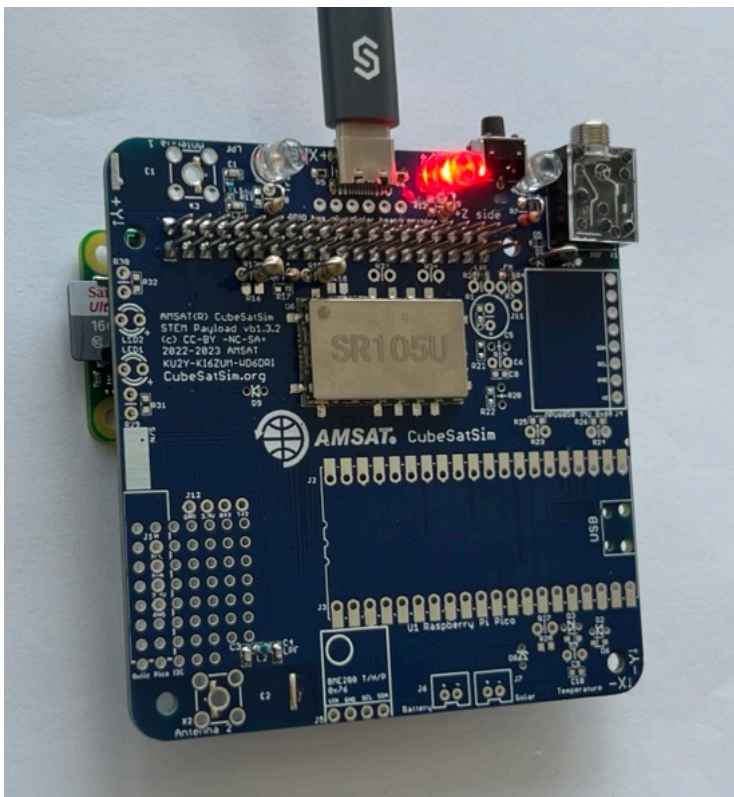
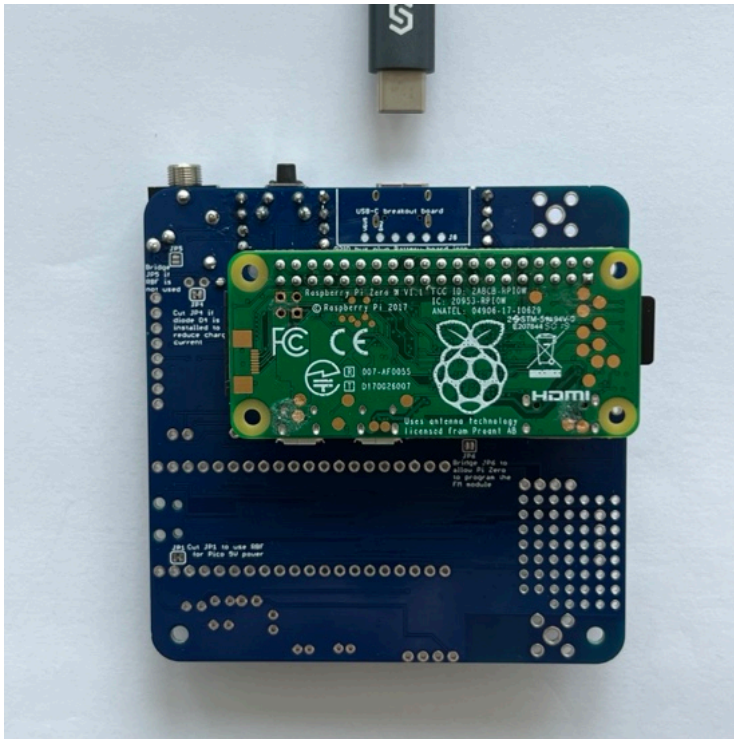


You can then insert the micro SD card in your Pi Zero 2:

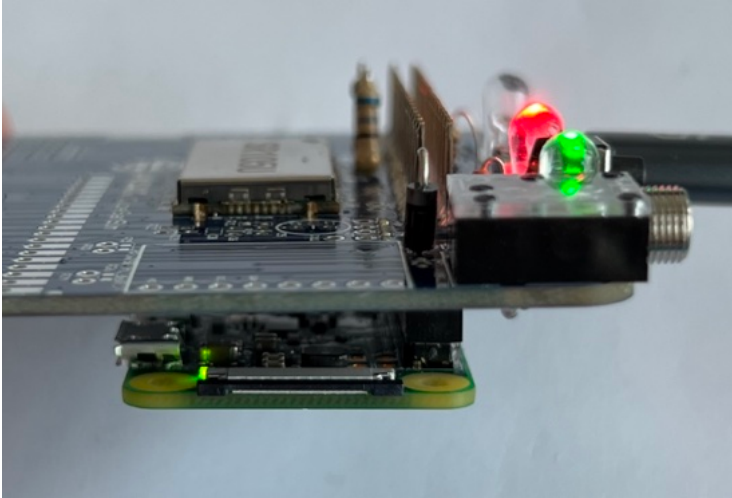


If you don't have your Main board, you can still test the software by plugging a micro USB cable into either micro USB connector on the Pi Zero 2. Without the Mainboard, it will only transmit the callsign in CW (Morse Code) once at the start. If you have a radio or ground station set up, you will hear it transmit at 434.9 MHz.

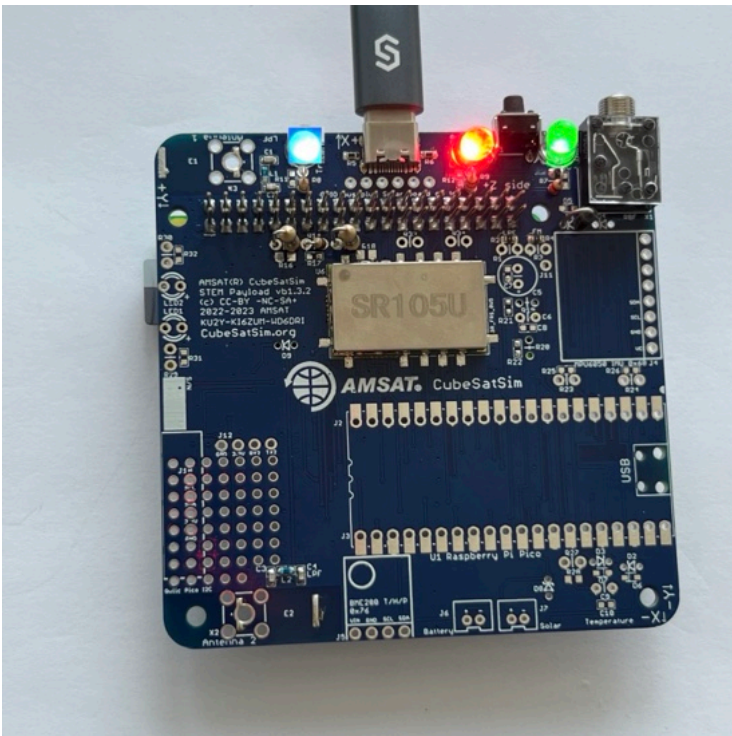
To fully test the software, plug your Pi Zero 2 into the Main board then plug the USB-C cable into the Main board. If you have your Main board built up to Step 1, you can plug the Pi Zero 2 into the bottom of the GPIO header.



The red LED will immediately illuminate. After 30 seconds, the green LED will also illuminate. You will also see the tiny green LED on the Pi Zero 2 blink occasionally:



Then, the blue transmit LED will turn on.



## Logging into your Raspberry Pi

---



You can customize your software by logging into the Pi. If you have an HDMI monitor (and mini HDMI adapter or cable) and a USB keyboard (and micro USB to USB adapter or OTG cable), you can access the console directly.

## Wi-Fi login

You will need to connect your Pi Zero 2 to your Wi-Fi so it can update the code from GitHub.

To get it on your Wi-Fi, follow these steps. First type this command:

```
sudo raspi-config
```

Then select `System Options` by hitting `Return`, then select `Wireless LAN` by hitting `Return` again. Then put your Wi-Fi name in the `SSID` field then hit `Return`, then type the Wi-Fi password then `Return` again. Hit the `Tab` key twice until `Finish` is highlighted, then hit `Return`. You may need to reboot. You can tell if you have Internet by typing:

```
timeout 10 ping amazon.com
```

If you see messages such as "64 bytes received" then you are online.

You can now update the Pi Zero 2 software by typing:

```
cd
```

```
CubeSatSim/update
```

You can also login remotely from your PC if it is on the same Wi-Fi as the Pi Zero. For example, in the Terminal Window or Windows Command Prompt, you can type:

```
ssh pi@cubesatsim.local
```

The first time you login, you will be prompted  
Are you sure you want to continue connecting  
(yes/no/[fingerprint])? Type yes then return to  
continue, then type the password then return.  
Note that the password will not be displayed as  
you type it.

## Ethernet over USB

If you don't connect your Pi Zero 2 to your Wi-Fi,  
you can access your Pi from a computer using  
just a USB cable. This Ethernet over USB service  
is known as RNDIS on Windows and USB  
Gadget on Linux.

On Mac or Linux, you don't need to install any  
software to use this. On Windows, you need the  
mDNS discovery program Bonjour. If you already  
have iTunes installed, you should have it. If not,  
you can download it from Apple here:

[https://support.apple.com/kb/dl999?  
locale=en\\_US](https://support.apple.com/kb/dl999?locale=en_US)

Note that it is called Bonjour Printer Services,  
since that is the most common standalone use  
for the application. Run the installer and reboot.

On your Pi Zero 2, connect a micro USB cable to  
the inner micro USB connector which is labeled  
"USB", not the one on the edge labeled PWR IN  
(Power In) as shown here:

(TBD)

When you connect the other end of the USB  
cable to your computer, it will power up the Pi.  
After a few moments, you will have network  
connectivity to your Pi.

For example, in the Terminal Window or Windows Command Prompt, you can type:

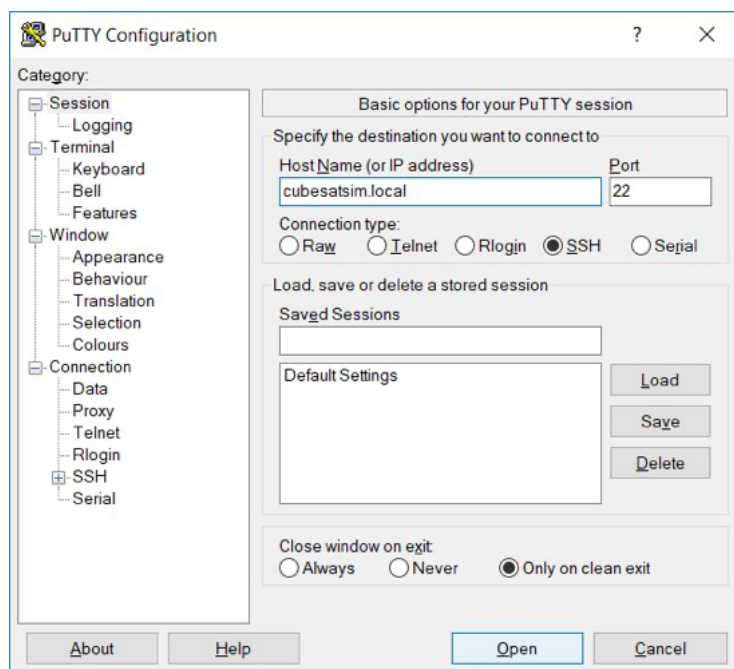
```
ssh pi@cubesatsim.local
```

and you can login using the default password raspberry.

Sometimes, this doesn't work on Windows - if you are unable to connect after installing the software, try the steps described here:

<https://fredicraab.wordpress.com/cubesat-simulators/raspberrypi-zero-windows-10-rndis-driver-problems/>

Alternatively, you can use a free application such as PuTTY on Windows. Download it here <https://www.putty.org/> and open it, setting the Host Name to `cubesatsim.local` and then click Open to connect:



Here is how it looks when you login on Windows using PuTTY after entering the default password raspberry

```
pi@cubesatsim: ~
login as: pi
pi@cubesatsim.lan's password:
Linux cubesatsim 5.4.79+ #1373 Mon Nov 23 13:18:15 GMT 2020 armv6l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

This Raspberry Pi image has the CubeSatSim software installed and runs automatic
ally as systemd cubesatsim.service. For more information see https://CubeSatSim.
org. To update to the latest version, enter this command:

CubeSatSim/update.sh

Last login: Sun Feb  7 13:47:32 2021 from 192.168.8.243

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

pi@cubesatsim:~$
```

You should set your amateur radio callsign which is transmitted as CW telemetry on 434.9 MHz (unless another frequency has been set using the `CubeSatSim/config -F` command) every time the CubeSatSim software starts up. You should edit the `CubeSatSim/sim.cfg` file to set your own callsign. You can also change localization settings or configure your local WiFi network by running `sudo raspi-config`.

The CubeSatSim software runs as a systemd service called `cubesatsim.service`. There is also a `transmit.service` which runs as well.

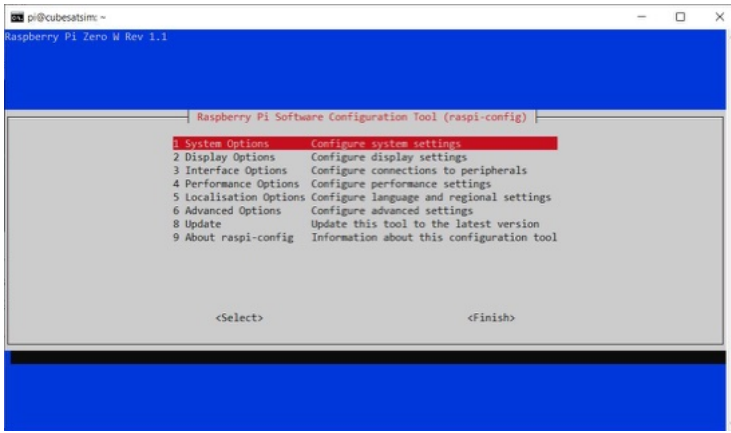
## Changing the hostname

If you want to change the hostname from `cubesatsim`, you can run `raspi-config` by typing this command:

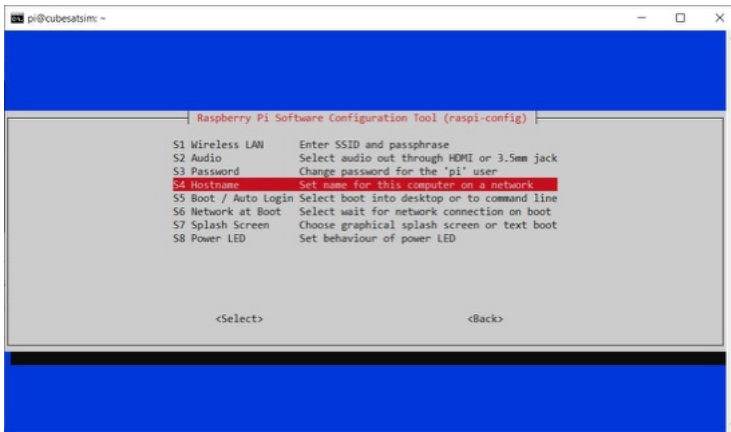
```
sudo raspi-config
```

You will see menu like this:

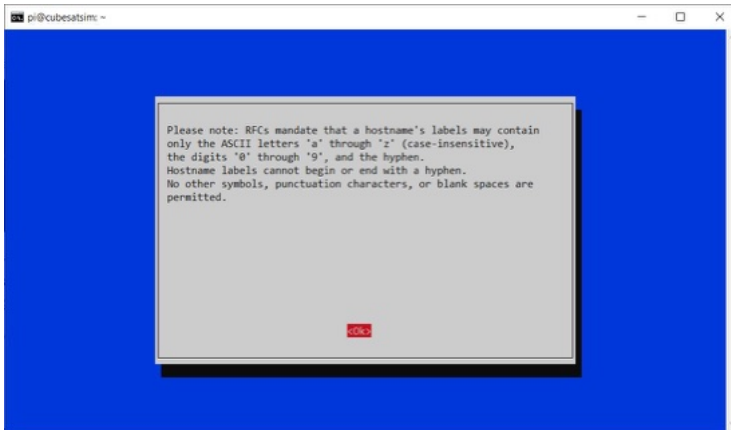




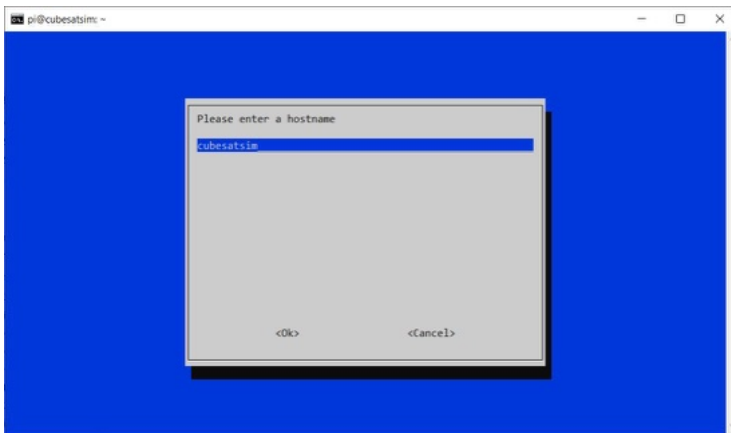
Hit the Enter key to select the System Options (or use the up and down arrows if you are on a different option). You will then see:



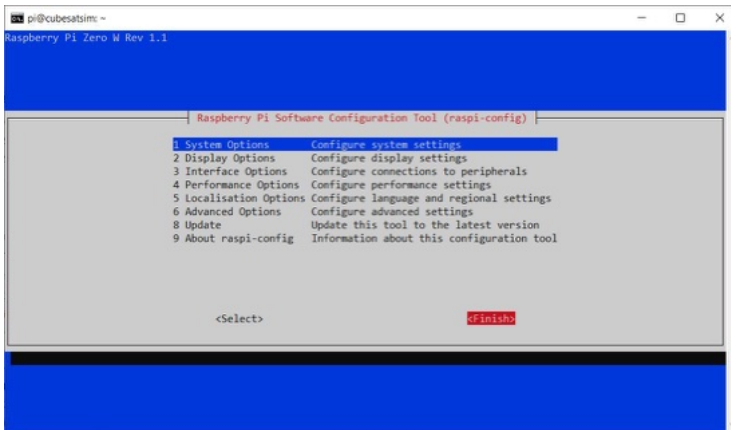
Use the down arrow to select Hostname then hit Enter. You will then see this message saying what characters can be used:



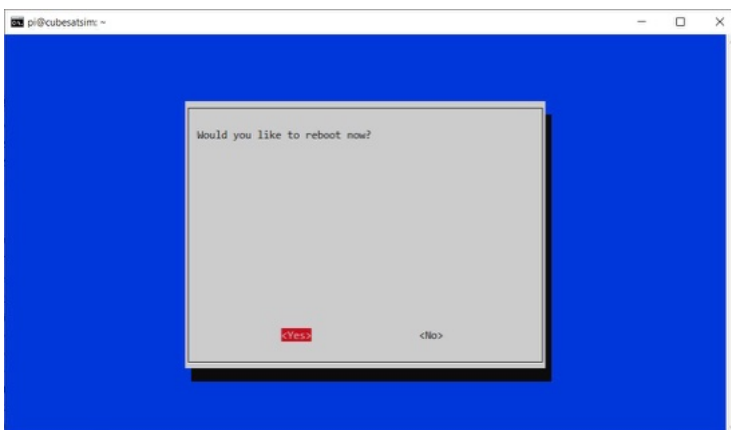
Hit the Enter key to continue:



Now enter your chosen hostname then hit Enter or use the down arrow to select the OK button then hit Enter.



Use the right arrow to select the Finish button then hit Enter:



Hit Enter to select Yes to reboot. When the Pi reboots in about 30 seconds, you will need to ssh using your new hostname.

# Installation Script Option Steps

---

You can start with the Bullseye version of Raspberry Pi OS. A Pi Zero or Pi Zero W should only run Lite while a Pi Zero 2 can run the Desktop. Make sure you create a user `pi` and do the install with that user or there may be issues with configuration files and settings as it references many paths in `/home/pi/*`. Your Pi will need to have internet access to update settings and install packages.

To get the software follow these steps:

```
sudo apt-get update && sudo apt-get dist-upgrade -y
```

```
sudo apt-get install -y git
```

```
git clone  
http://github.com/alanbjohnston/CubeSatSim.git
```

```
cd CubeSatSim
```

You are now ready to install the software.

```
./install
```

You will be prompted for your amateur radio callsign. If you don't have one or just hit Return, the default AMSAT will be used. The Callsign is transmitted as a CW ID (Morse code) each time the CubeSatSim software starts up.

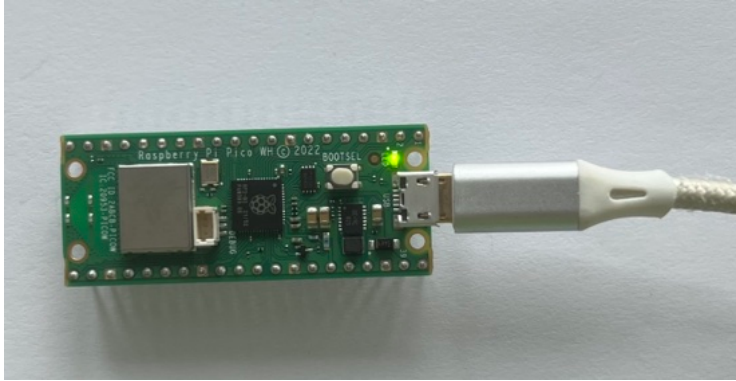
The installation script will run for quite a while. It will prompt you if you want to modify `/boot/config.txt` file. Type a `y` and the script will complete. You will need to reboot.

## Raspberry Pi Pico Software

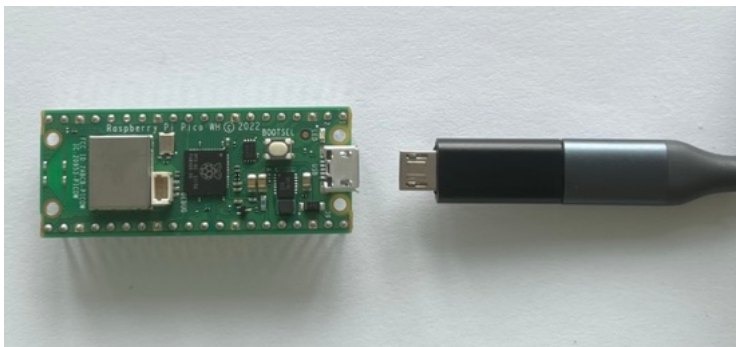
---

These instructions are for the CubeSatSim software for the Raspberry Pi Pico processor.

You will need a Raspberry Pi Pico W (a Pico works as well but doesn't have WiFi for future features) and either a micro USB cable:



Or a USB-C cable with a micro USB adapter:



They include how to install the software, check the software log, and do various debugging.

## Software installation

You will need:

- Raspberry Pi Pico or Pico W microcontroller
- micro USB cable to connect to your computer
- Computer with Arduino IDE running



The simplest way is to install the software image, as described below. If you want to compile the Pico code yourself and make modifications or add sensors, Here's how to [compile the code and add more sensors](#).

Here are the steps to install the software image:

- Download the UF2 file with the latest software from here onto your computer: Go to <https://github.com/alanbjohnston/CubeSatSim/releases/> and select the latest one. In the notes for the release, you will find a link to the .UF2 file (it is also in the Assets, but you have to click on it to see it). Download the UF2 file
- While holding down the BOOTSEL button (white button on the Pico) plug your Raspberry Pi Pico or Pico W into your computer using the micro USB port
- It should mount as a flash drive called RPI-RP2. Drag the .UF2 file you downloaded previously to the Pico flash drive
- After a few seconds, the LED on the Pico should flash on and off about the once per second.

## Viewing Software Logs

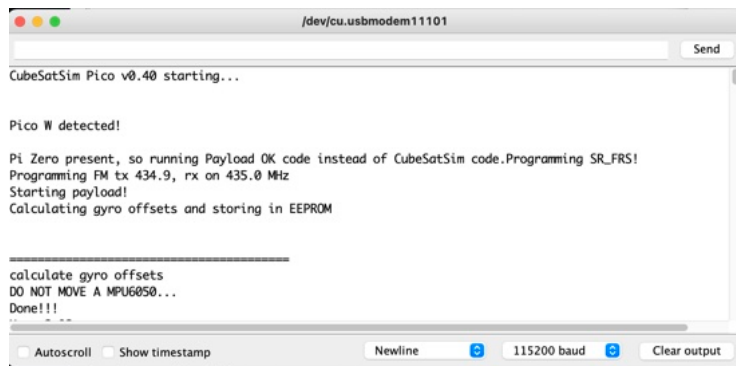
The software continuously writes a log file to the serial port. You can view it on a Computer with the Arduino IDE by running the Serial Monitor.

The board selected doesn't really matter. It could be the Raspberry Pi Pico or Pico W if you have it installed. If you don't, you could just select the Arduino Uno board.

You will need to select the Port after you plug the Pico in with the micro USB cable to your computer.

Click on the Serial Monitor icon to open it. You should set the baud rate to 115200 using the drop down menu.

Here's what you will see from the start when the Pico boots up not plugged into anything (the version number may be slightly different):



```
CubeSatSim Pico Payload v0.2
starting...
```



```
Pico W detected!
```

```
Pi Zero present, so running Payload OK
code instead of CubeSatSim
code. Programming SR_FRS!
```

```
Starting payload!
Calculating gyro offsets and storing
in EEPROM
```

```
=====
calculate gyro offsets
DO NOT MOVE A MPU6050...
Done!!!
X : -2.92
Y : -0.64
Z : -4.18
Program will start after 3 seconds
=====
/payload.cfg file
```

```
Writing string
-3.027673 5.815880 -1.333720
OK BME280 26.15 991.59 181.94 31.96
MPU6050 -0.07 0.00 0.04 -0.00 -0.06
0.98 XS2 0.0000 0.0000 0.00 MQ 170
Squelch: 1
OK BME280 26.15 991.62 181.65 31.87
MPU6050 -0.07 -0.33 0.04 0.00 -0.06
0.99 XS2 0.0000 0.0000 0.00 MQ 170
Squelch: 1
OK BME280 26.15 991.63 181.56 31.83
MPU6050 0.16 -0.18 0.02 0.00 -0.06
0.99 XS2 0.0000 0.0000 0.00 MQ 170
Squelch: 1
```

If you see this, you have successfully installed the CubeSatSim software on the Raspberry Pi Pico processor.

Next step is to [setup your Ground Station](#).

+ Add a custom footer

# 3. Ground Station

Edit New page

Alan Johnston edited this page 3 weeks ago · [34 revisions](#)

If the images on this page fail to load, you can [download a PDF of this page here](#).

## 3. Ground Station

The Ground Station for the CubeSatSim is used to receive the radio transmissions so you can listen to them and decode telemetry.



### Checklist

The BOM has a sheet "By Steps" which lists the parts needed for each step in order. <http://cubesatsim.org/bom> If you have a Google account, you can make a copy of this spreadsheet ("File" then "Make a Copy") and check off each part as you install it.

For example, here is the checklist for this step:

- ▼ Pages 147
  - Find a page...
  - ▶ [Home](#)
  - ▶ [1. Main Board 1](#)
  - ▶ [2. Software Install](#)
  - ▼ [3. Ground Station](#)
    - 3. Ground Station
      - Checklist
      - 3.1 Raspberry Pi Ground Station
        - Video
        - Fox-in-a-Box v3 Image
        - 3.1 PC Ground Station
  - ▶ [4. Main Board 2](#)
  - ▶ [5. Battery Board](#)
  - ▶ [6. Solar Board](#)
  - ▶ [7. Solar Panels and Frame](#)
  - ▶ [8. Board Stack](#)
  - ▶ [9. Final Testing](#)
  - ▶ [Adding New Sensors](#)




<input type="checkbox"/>	Step 3. Ground Station			<a href="https://github.com/alanbjohnst">https://github.com/alanbjohnst</a>
<input checked="" type="checkbox"/>	<b>Item</b>	<b>Qty</b>	<b>Location</b>	<b>Image</b>
<input type="checkbox"/>	Alt 6a: Your PC with RTL-SDR	1		
<input type="checkbox"/>	Alt 6a: SMA Antenna for 433 MHz right angle	1		
<input type="checkbox"/>	Alt 6b: Raspberry Pi 4 ground station	1		
<input type="checkbox"/>	Alt 6b: Pi 4 power supply	1		
<input type="checkbox"/>	Alt 6b: micro SD card 16 GB with FIABv3	1		
<input type="checkbox"/>	Alt 6b: RTL-SDR	1		
<input type="checkbox"/>	Alt 6b: SMA Antenna for 433 MHz right angle	1		
<input type="checkbox"/>	Alt 6b: 7" touchscreen with speakers and star	1		
<input type="checkbox"/>	Alt 6b: Wireless mouse	1		
<input type="checkbox"/>	Alt 6b: USB extender cable	1		
<input type="checkbox"/>	Alt 6b: right angle USB-C adapter	1		

- ▶ [Command and Control](#)
  - ▶ [Creating the CubeSatSim Raspberr...](#)
  - ▶ [CubeSatSim Lite](#)
  - ▶ [CubeSatSim Loaner User Guide](#)
- Show 132 more pages...

+ Add a custom sidebar

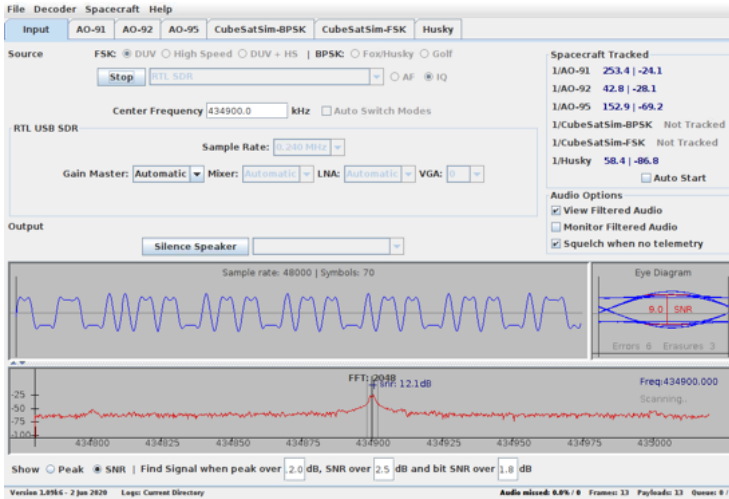
The Ground Station for the CubeSatSim utilizes FoxTelem, the open source AMSAT telemetry decoding software by Chris Thompson, G0KLA/AC2CZ which you can download here <https://www.amsat.org/foxtelem-software-for-windows-mac-linux/>

Clone this wiki locally

<https://github.com/alanbjohnston> 

You will need to install two "spacecraft" files in FoxTelem for the CubeSatSim in order to decode telemetry.

For more information on how FoxTelem works and its spacecraft files see [https://www.g0kla.com/foxtelem/amsat\\_telemetry\\_designers\\_handbook.pdf](https://www.g0kla.com/foxtelem/amsat_telemetry_designers_handbook.pdf)

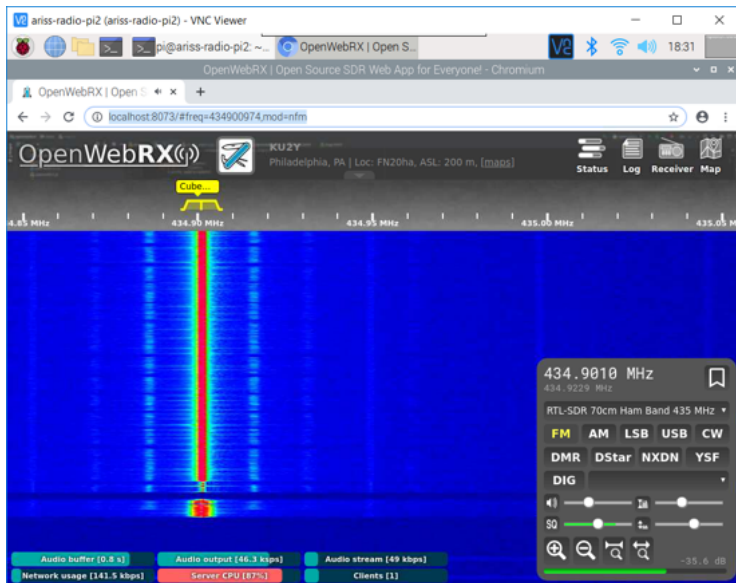


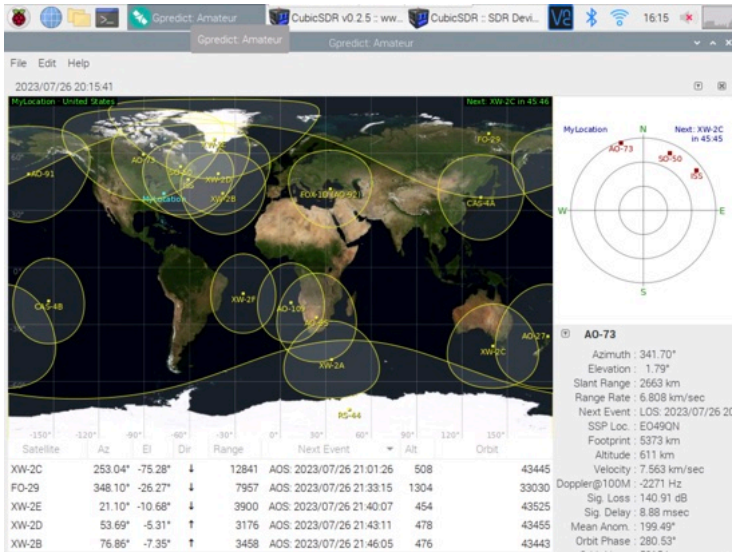
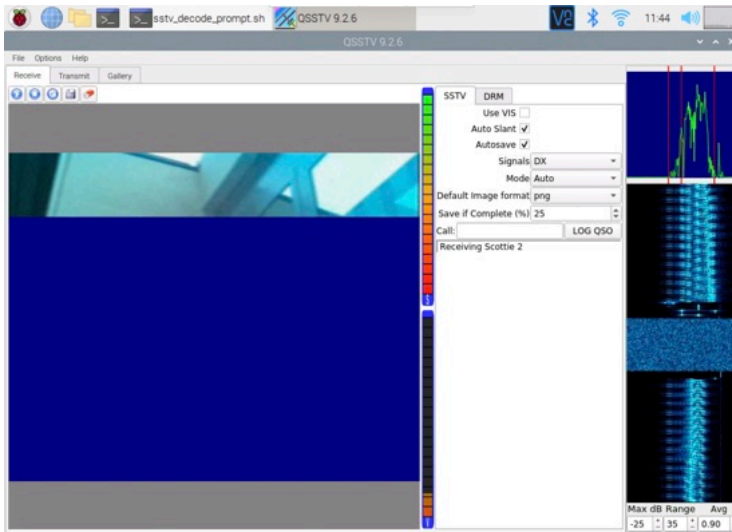
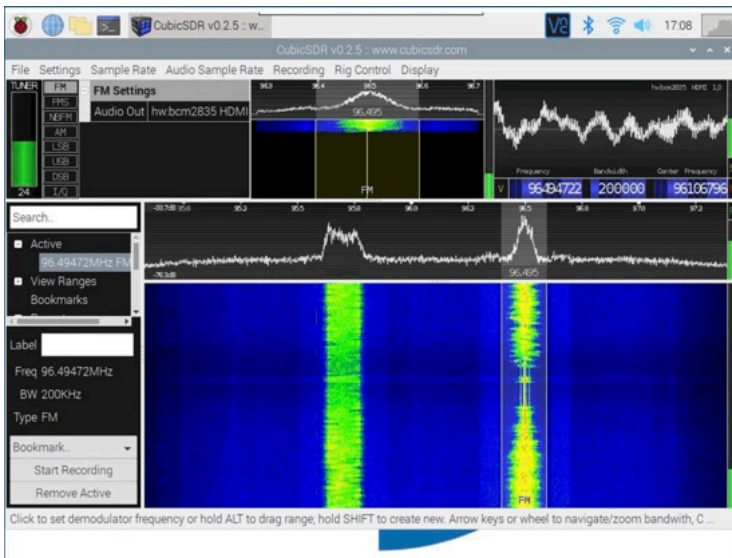
You will need an SDR such as the RTL-SDR. I recommend this one <https://www.amazon.com/gp/product/B0129EBDS2>. You will need a 433 MHz antenna such as <https://www.digikey.com/en/products/detail/rf-solutions/ANT-STUBR-433SM/5845732>. Or, you can order an RTL-SDR with a telescopic dipole antenna.

For the CubeSat Simulator Lite, older versions of the CubeSatSim (vB3 and older) or the current version running in AFSK 1200 bps APRS telemetry mode, these are the [Ground Station instructions](#).

The simplest way to get a fully functional Ground Station is to use a Raspberry Pi 3B or 4B with an RTL-SDR dongle and download the SD card image as described in the next section.

Besides FoxTelem, the ARISS Radio Pi image also has OpenWebRX, CubicSDR, QSSTV, and Gpredict





### 3.1 Raspberry Pi Ground Station

Video

Here is a video of how to get the Raspberry Pi Ground Station up and running:

[https://youtu.be/l1o9LD\\_2hIU](https://youtu.be/l1o9LD_2hIU)

Here is a playlist of videos on how to use all the software: [https://youtube.com/playlist?list=PL-pgyk1pc4\\_OBQOov45FaBZXXWYZh\\_oLN](https://youtube.com/playlist?list=PL-pgyk1pc4_OBQOov45FaBZXXWYZh_oLN)

## Fox-in-a-Box v3 Image



You can write a pre-built Raspberry Pi Ground Station disk image to your SD card. Your Pi 3B or 4B would then boot up immediately running FoxTelem software. This image is based on the [ARISS Radio Pi](#). It has FoxTelem to decode FSK and BPSK telemetry, and also QSSTV to decode SSTV. The Web SDR using OpenWebRX Web will decode the AFSK telemetry by selecting Digital/Packet demodulation.

You will need a micro SD card with at least 16GB of capacity and a micro SD card writer and free software such as balenaEtcher:

<https://www.balena.io/etcher/> You will need to have administrator privileges on your computer in order to do this.



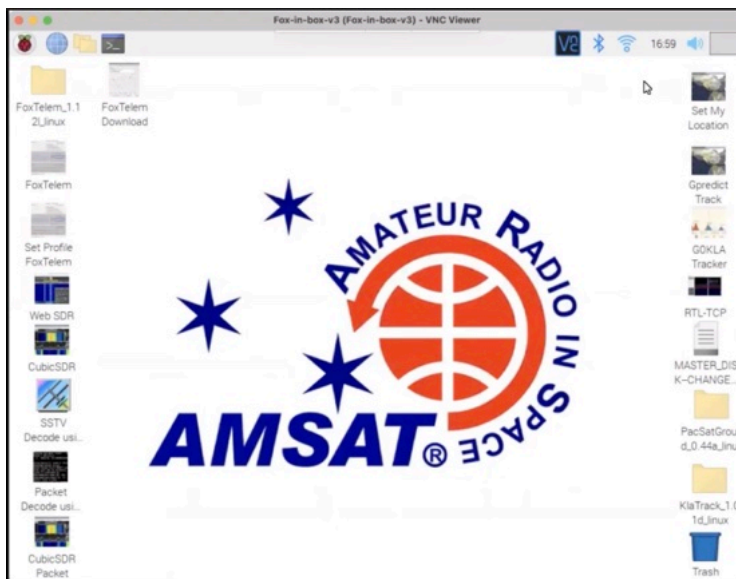
Download the disk image here using your web browser: <https://cubesatsim.org/download/Fox-in-box-v3.iso.gz>

Run balenaEtcher and select the Flash from File option, then select the image file you downloaded.

Insert your micro SD card - the card will be erased in the process of writing. Click on Select Target and select your SD card writer.

Then select Flash to begin writing. You will need to input your administrator password, then the write will begin. It will take a while.

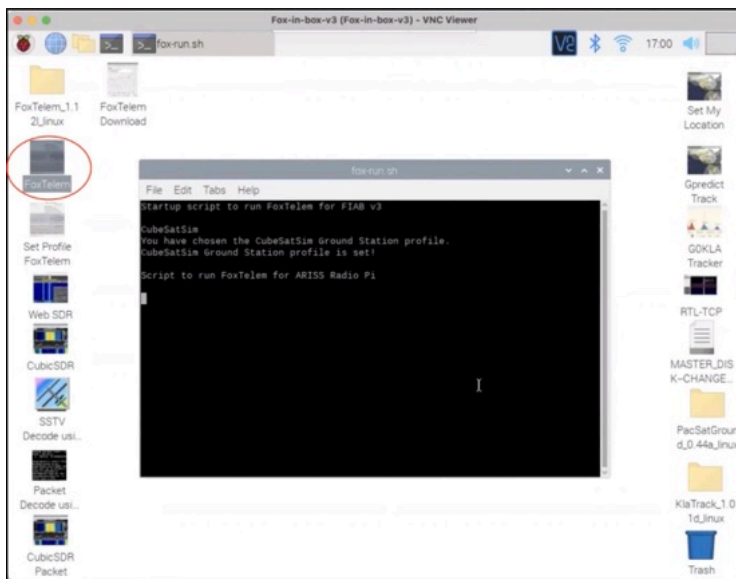
You can then insert the micro SD card in your Pi 3B or 4B, plug in your RTL-SDR dongle, and it will boot. You should see this Pi Desktop:



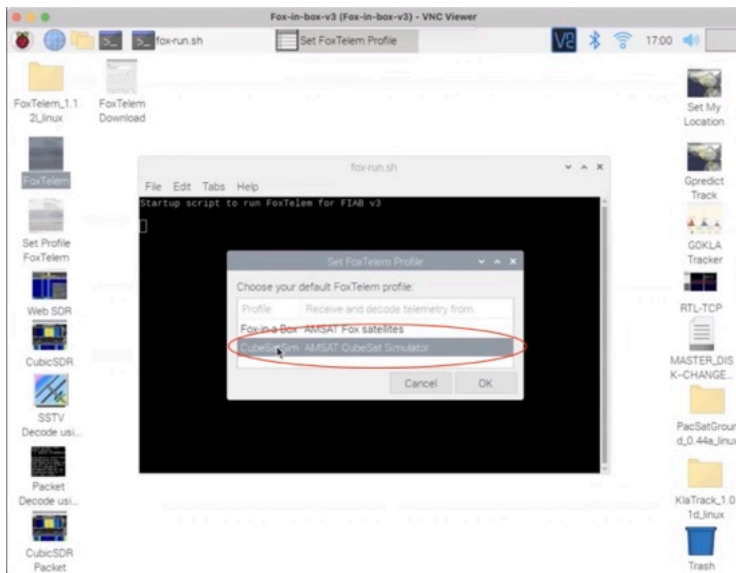
This video shows the basic operation of the Pi Ground Station:

<https://www.youtube.com/watch?v=68ULJdP5OZw>

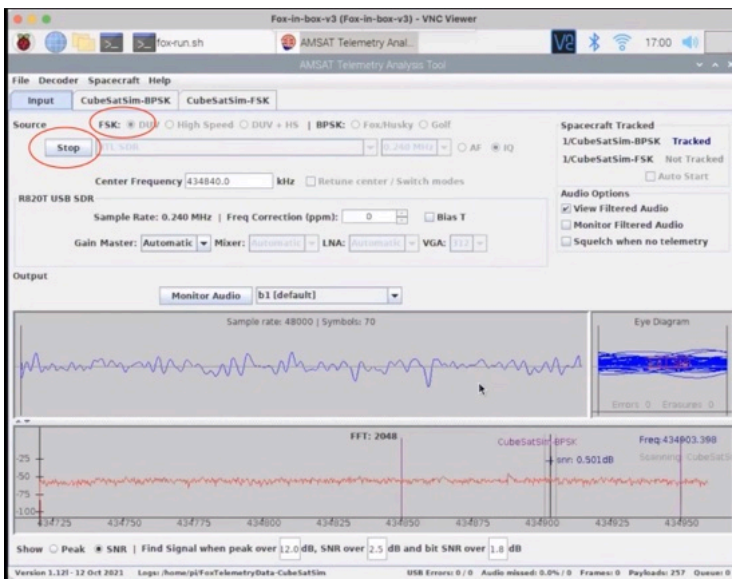
Double click on the FoxTelem icon on the Desktop (2nd from the top on the left - circled in red below) and a script will start to run:



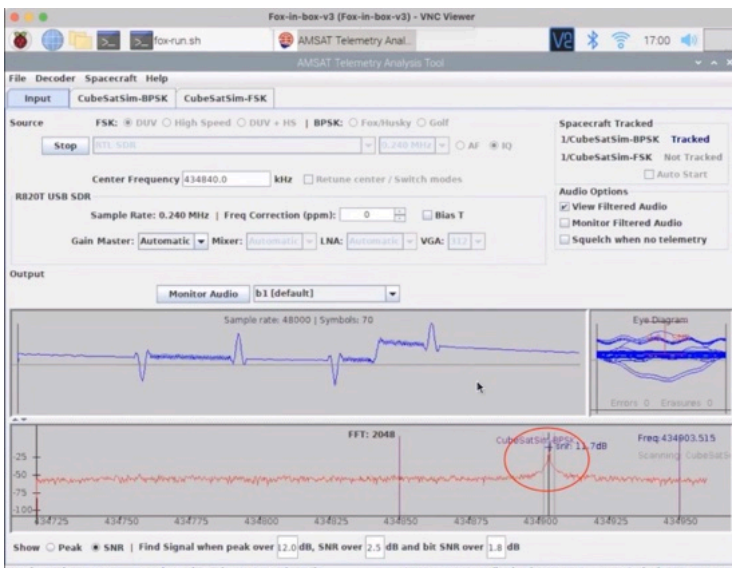
The first time you run FoxTelem, you will be prompted to choose your profile - select CubeSatSim AMSAT CubeSat Simulator, shown circled in red here:



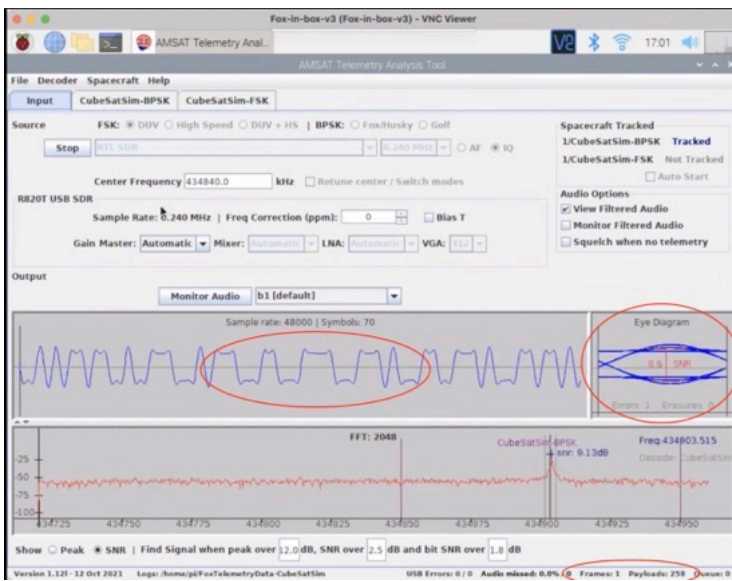
From now, on, The when you boot up your Pi, it will auto run FoxTelem. FoxTelem will open and be in the mode you were running when last quit the program. If you are running it for the first time, it might look like this:



If it doesn't, you might have to click on the FSK DUV radio button or the Start button, circled in red in the image above. This image shows no signal being transmitted by the CubeSatSim. If the CubeSatSim just powered up, it might be transmitting the CW ID, in which case you might see this:

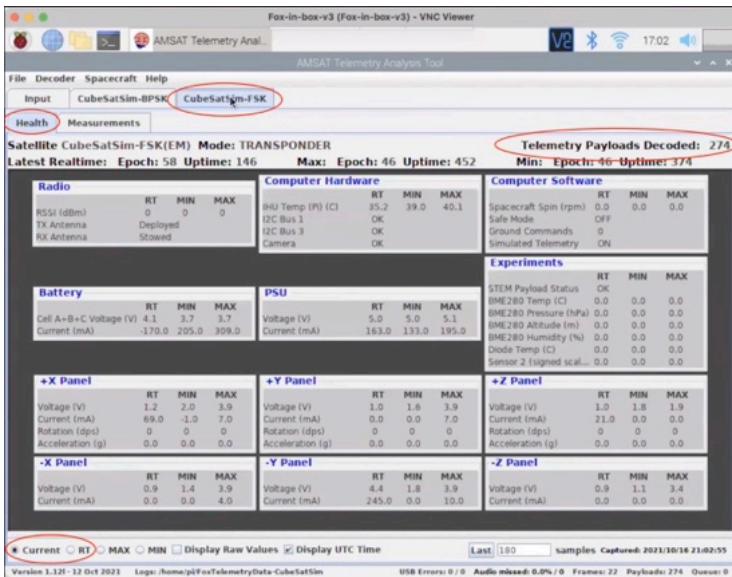


In the FFT window at the bottom of the screen, you can see a peak indicating a transmitted signal from the CubeSatSim, which is circled in red in the above image. Within 30 seconds, if the CubeSatSim is transmitting in FSK mode (mode 2), you should see this:



In this image, the digital waveform and the eye diagram are circled in red. At the bottom right of the screen, you should see the Payloads and Frame counts increasing every 4 seconds or so.

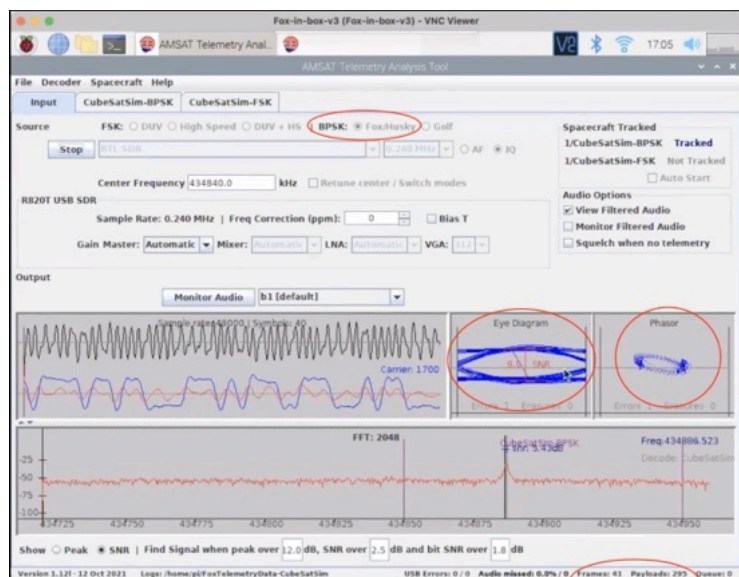
To see the telemetry data, click on the CubeSatSim-FSK tab and the Health tab, circled in red here:



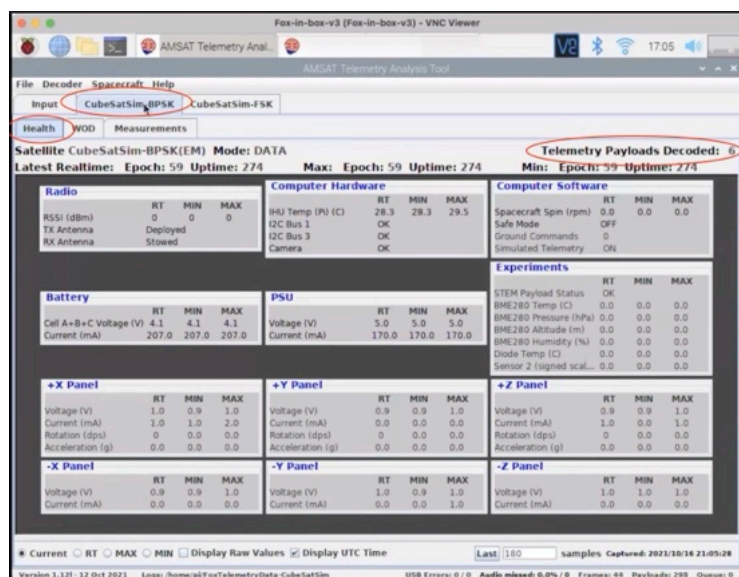
At the top right, the Telemetry Payloads Decoded count should be increasing. If this data isn't changing, make sure the Current radio box at the bottom is selected, indicating that you are seeing live data.



If you change the CubeSatSim to BPSK mode (mode 3), you can decode that telemetry in FoxTelem by clicking on the Input tab, then clicking the Stop button, then the BPSK: Fox/Husky radio button, then the Start button. If the CubeSatSim is transmitting in BPSK mode, you should see:



You will see a Phasor and Eye Diagram as shown here, and the Frame and Packet counts will increase. To see the telemetry data, click on the CubeSatSim-BPSK tab and you should see data:



### 3.1 PC Ground Station

If you are using an RTL-SDR dongle, I'd recommend first installing SDR# (pronounced SDR Sharp) so you can test your RTL-SDR. Then install FoxTelem.

Here are the instructions: <https://rtl-sdr.com/qsg>

I'd also recommend SDR++ as an SDR as well: <https://github.com/AlexandreRouma/SDRPlusPlus/releases>

You can also install FoxTelem on your computer and use it as your Ground Station.

For more detailed instructions and activities, see the AMSAT Journal article <http://cubesatsim.org/content/CubeSatSimPaper6.pdf>

Go to <https://www.amsat.org/foxtelem-software-for-windows-mac-linux/> and download the highest version number of FoxTelem for your OS (Windows, Mac, or Linux - for Raspberry Pi). Follow the install instructions and run that version of FoxTelem. You may have to update your version of Java – follow the prompts to do so.

When FoxTelem runs for the first time, a box saying "Welcome to the Amsat Fox Telemetry Analysis Tool" will pop up. Select Continue to choose the default location for your directory. Select Yes a number of times to install the spacecraft files. In a minute, the FoxTelem application will load.

You should see a CubeSatSim-FSK tab and a CubeSatSim-BPSK tab - this is because we use different Fox ids for the two different telemetry modes. The FSK mode emulates Fox-1A, 1B, 1C, and 1D telemetry, while the BPSK mode emulates Fox-1E and HuskySat-1 telemetry.

To test FoxTelem, download this WAV file and save:

<https://cubesatsim.org/wav>

In the Input tab, under FSK select the DUV option. On the audio source, select the AF option. Next, click on the "Select audio source here the press start" drop down menu and select "Load Wav File". A file menu will open and you should select the WAV file you downloaded earlier, then click on the Start button. You should get several Frames and Payloads decoded (as shown at the bottom of the window). Select the CubeSatSim-FSK tab and you should see telemetry data displayed. You can click on a value to get a graph.

If you have an RTL-SDR USB dongle plugged in, you can receive live telemetry from your CubeSatSim. In the Input tab, select "RTL SDR" with FSK DUV and IQ selected. Under Settings click the box next to Find Signal for automatic tuning.

Set the Center Frequency as 434840 and then click Start. NOTE: The CubeSatSim transmits telemetry at 434.9 MHz (+/- 15 kHz) but with an SDR you don't want to tune to this exact frequency. Drag the bottom of the FoxTelem window down and the FFT spectrum plot window will appear, or go under the Decoder menu and check the Show FFT option. You should see a telemetry signal from your CubeSat Simulator in the spectrum at the bottom of the screen.

Click on the signal in the FFT window so it is centered, and if your CubeSatSim is transmitting DUV FSK telemetry, you should see the Frames and Payloads counts increment in the bottom right corner of window.

Click on the CubeSatSim-FSK tab.

At the top right, you should see Telemetry Payloads Decoded showing the number of payloads you have decoded.

Click on the Computer Temperature value to see a real-time graph.

The time axis shows the Uptime, the time in seconds since the Raspberry Pi was last rebooted, and the Resets, a count of the number of times the software has been run on the Pi.

In the Input tab, click Stop. Click yes to the popup asking if you want to Stop decoding while pass in progress.

Select BPSK Fox/Husky then click Start.

To decode BPSK frames, the CubeSatSim will need to be in the BPSK mode.



Watch for Frames and Payloads counts to increase in bottom left corner of window.

Click on the CubeSatSim-BPSK tab and you will see telemetry.

Click on the value to see a real-time graph.

The next step is to [continue building the Main Board](#).

Back to the [CubeSat Simulator Wiki Home](#)

+ Add a custom footer

# 4. Main Board 2

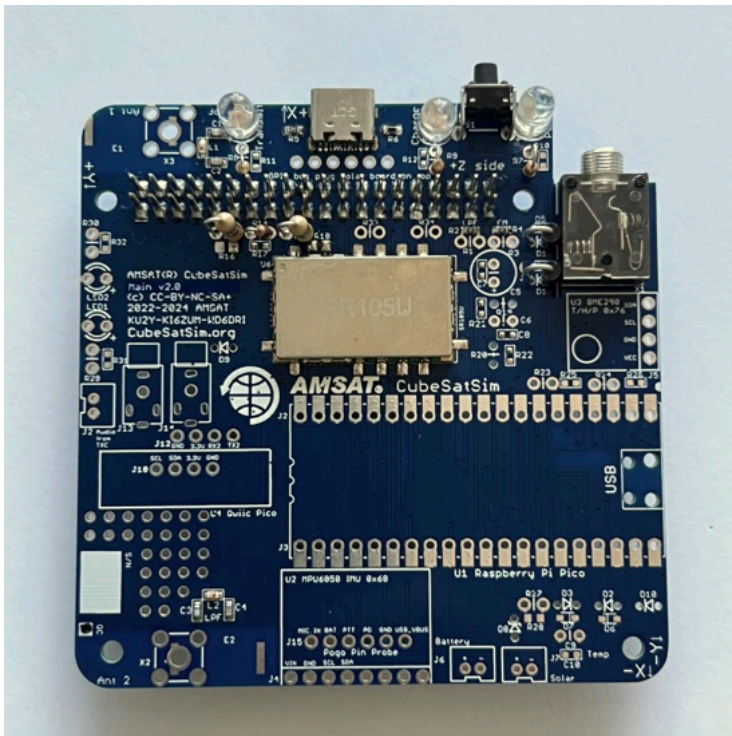
[Edit](#) [New page](#)

Alan Johnston edited this page 4 hours ago · [40 revisions](#)

If the images in this page fail to load, you can [download a PDF of this page here](#).

## 4. CubeSatSim Main Board Assembly Part 2

After [Part 1](#), here is the Main board:



We can now start doing some intermediate tests on the board.

You will need these tools:

- Pages 147
- Find a page...
- Home
- 1. Main Board 1
- 2. Software Install
- 3. Ground Station
- 4. Main Board 2
  - 4. CubeSatSim Main Board Assembly Part 2
    - Checklist
    - Schematic
  - 4.1 Testing
    - Video
    - Pi Blink Test
    - Power LED Test
    - Transmit LED Test
    - Transmit Test
    - Push Button Test
    - RBF Switch Test
  - 4.2 Build
    - Video
    - Push Button Test
    - Continue Build
  - 4.3 Antenna Install

- Safety glasses (to protect eyes while soldering or trimming leads)
- Soldering iron and solder (I use lead-free solder, but leaded solder is easier to work with)
- Needle nose pliers (to bend leads and hold parts)
- Side cutters (to trim leads)
- glue, such as super glue to attach the JST connectors to the PCB

Other tools that are helpful:

- [Blue mounting putty](#) (to hold components in place while soldering)

The first part of this step is testing while the second part continues the assembly

## Checklist

Here is the Bill of Materials:

<https://CubeSatSim.org/bom>

The BOM has a sheet "By Steps" which lists the parts needed for each step in order. If you have a Google account, you can make a copy of this spreadsheet ("File" then "Make a Copy") and check off each part as you install it.

For example, here is the checklist for this step:

SMA Antennas

Tape Measure Dipole Antenna

Tape Measure Monopole Antenna

### 4.4 Testing

Video

Payload Sensor Testing

Complete Payload Test

Payload Serial Connection Troubleshooting

Adding Additional Sensors

Additional Sensor Info

▶ [5. Battery Board](#)

▶ [6. Solar Board](#)

▶ [7. Solar Panels and Frame](#)

▶ [8. Board Stack](#)

▶ [9. Final Testing](#)

▶ [Adding New Sensors](#)

▶ [Command and Control](#)

▶ [Creating the CubeSatSim Raspberr...](#)

▶ [CubeSatSim Lite](#)

▶ [CubeSatSim Loaner User Guide](#)

































Show 132 more pages...

+ Add a custom sidebar

Clone this wiki locally

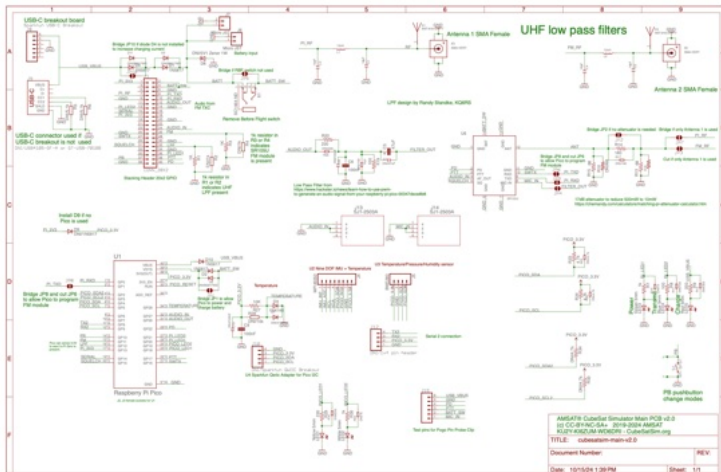
<https://github.com/alanbjohnston>



Ref	Item	Qty	Location	Image
	STEM Payload Board from Step 1	1		
	PI Zero 2 with SD card from Step 2	1		
	USB-C cable and power plug	1		
	RBF keychain	1		
RBF	3.5mm plug	1		
C5	47uF electrolytic capacitor	1	Top	
C6	100nF capacitor	1	Top	
R20	220 Ohm resistor	1	Top	
R19	100 Ohm resistor	1	Top	
J2, J3	20 pin female socket for Pico	2	Top	
R23, 24	4.7k resistor	2	Top	
	MPU6050 9 Axis Gyro GY-521	1	Top	
J4	Female 1x8 header	1	Top	
	BME280 board Temperature/Humidity/Pressure	1	Top	
J5	Female 1x4 header	1	Top	
R27	10k resistor	1	Top	
C9	100nF capacitor	1	Top	
D2	1N5817 diode	1	Top	
D3	1N4148 diode	1	Top	
J6	DT 2.0 connector	1	Top	
LED1	Green LED	1	Top	
R29	1k resistor	1	Top	
LED2	Blue LED	1	Top	
R30	100 Ohm resistor	1	Top	
J10	QWICC connector	1	Top	
	1x4 male breakout header	1	Top	
J13	2.5mm audio jack	1	Top	
X2, X3	Alt 2a: SMA vertical female connector	2		
	Alt 2a: SMA 6" male to female for antenna	2		
	Alt 2a: SMA 433 MHz antenna	2		
	Alt 2b: Tape Measure for antenna	1		
	Alt 2b: Nylon nut	1		
	Alt 2b: 2.5mm screw for tape measure antenna	2		

## Schematic

Here's the schematic for the v2.0 Main board for reference:



[https://github.com/alanjohnston/CubeSatSim/blob/42b777577a17e1c3bdc2f7bfbf3d5b9bec08124d/hardware/v2.0/cubesatsim-main-v2.0\\_schematic.pdf](https://github.com/alanjohnston/CubeSatSim/blob/42b777577a17e1c3bdc2f7bfbf3d5b9bec08124d/hardware/v2.0/cubesatsim-main-v2.0_schematic.pdf)

## 4.1 Testing

## Video

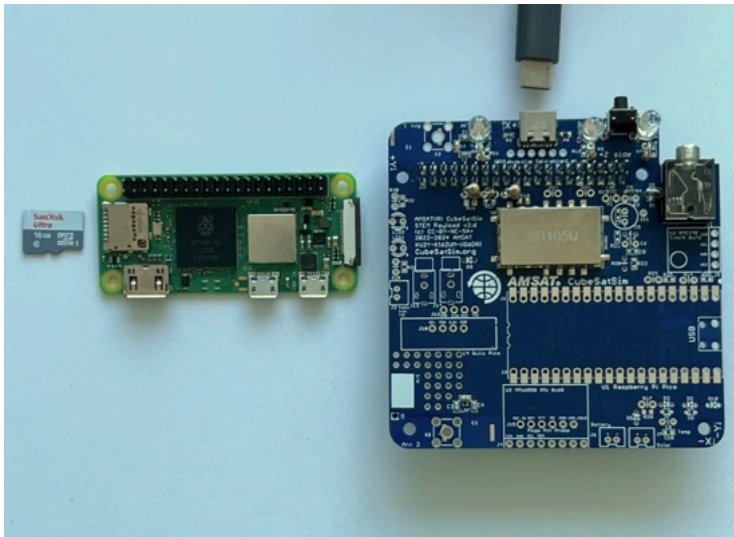


Here is a [video of the testing part of this step](#).

## Pi Blink Test

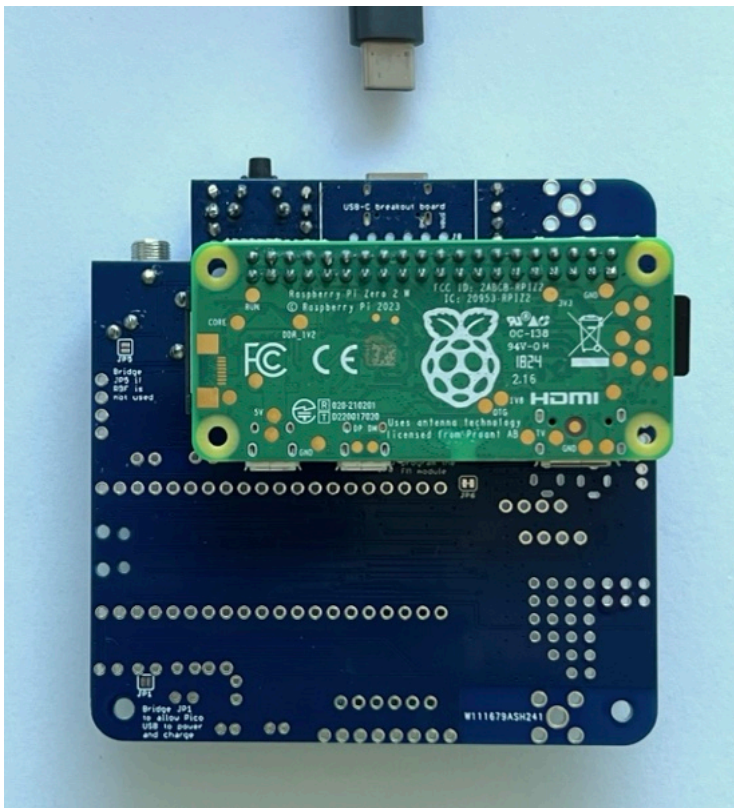
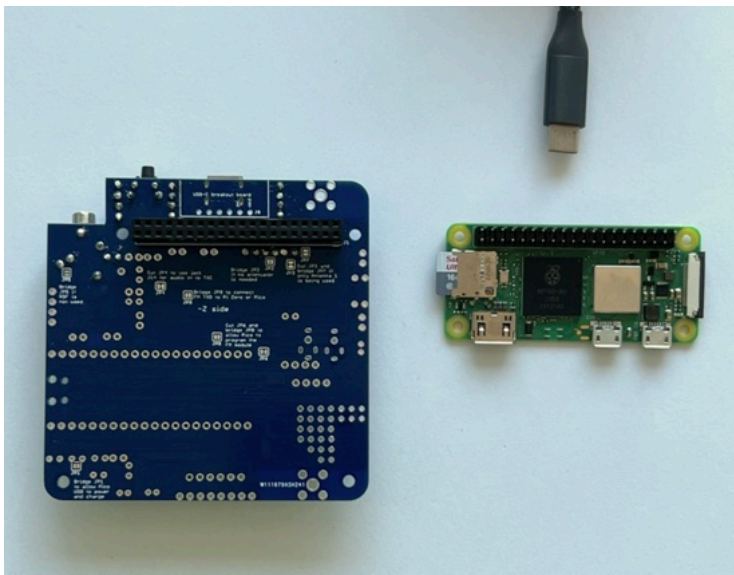
---

For these tests, you will need your Raspberry Pi Zero 2 or Pi Zero, a micro SD card with the CubeSatSim software installed (see the [Software Install instructions](#)), and the USB-C power cable (or a micro USB power cable or adapter if you power it directly to the Pi).

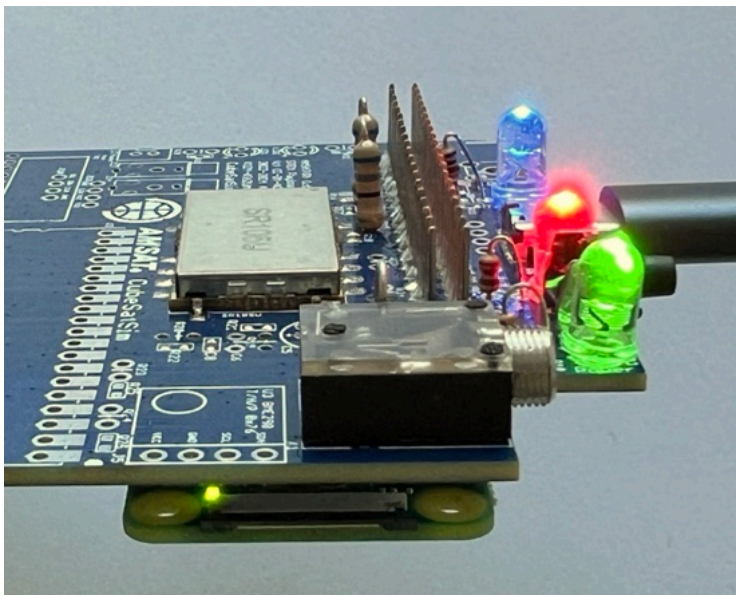


If your Pi Zero 2 does not have the 2x20 pin header installed, you will need to carefully solder it in. The pins should stick up on the side that has the ICs and the SD card holder.

Flip the Main Board upside down and plug the Pi Zero 2 into the bottom of the board so that it looks like this:

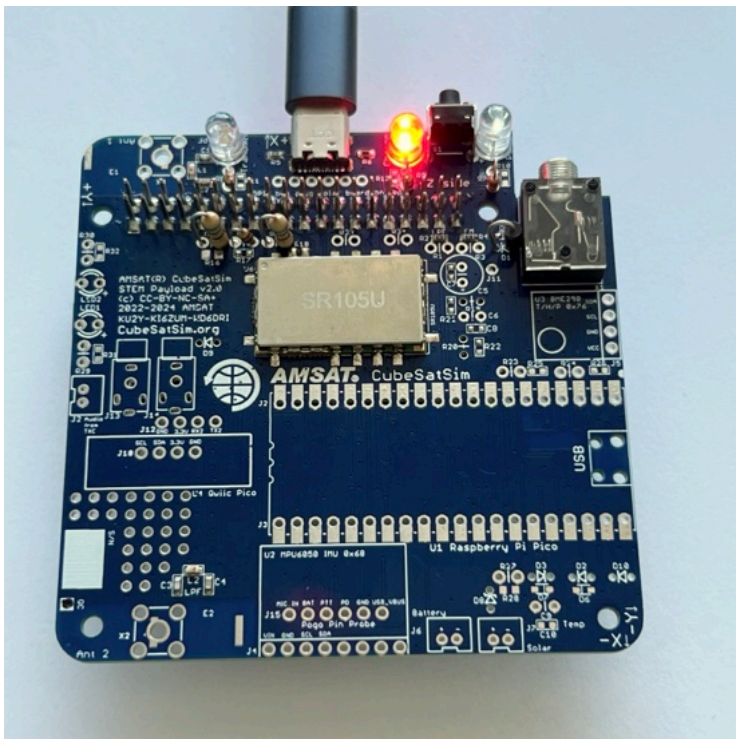


Plug your USB-C power cable directly via the PCB USB-C port. The Pi will turn on and the tiny green LED on the Pi will be on or blinking. Note that the LED is on the top of the Pi Zero 2 board, but you can see it through the lower left mounting hole, or from the side:



If you don't see this, make sure you have your micro SD card with the CubeSatSim software installed and the USB cable providing power to the Pi.

With the USB-C cable plugged into the board, the Red LED will be illuminated:

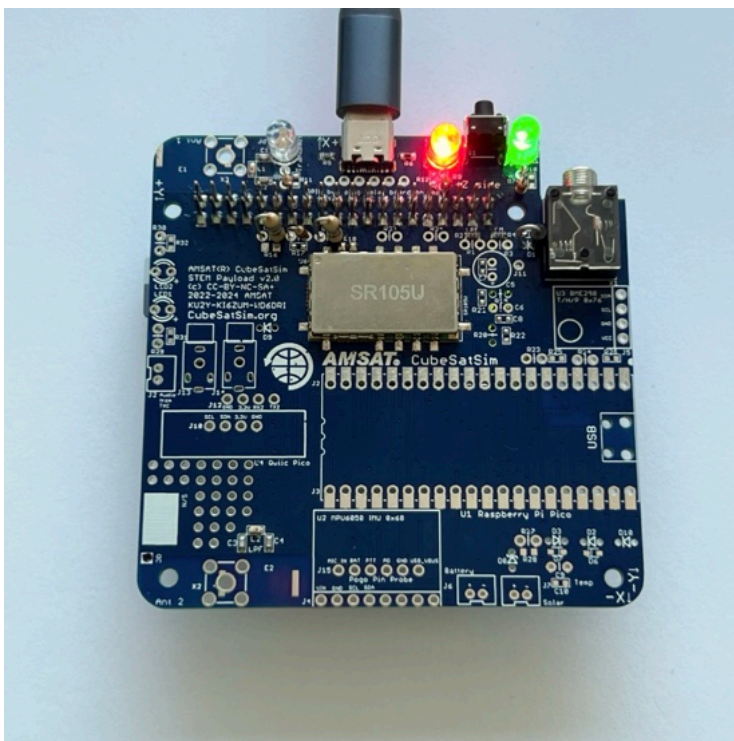


If the LED doesn't illuminate, it might mean there is a problem with the red LED LED5 or resistor R9 or the USB-C connector J9 or resistors R5 or R6.

## Power LED Test

---

With the Pi plugged into the main board and the power cable plugged into the Pi micro USB port, about 30 seconds after booting, the CubeSatSim software will run and turn on the green Power LED.



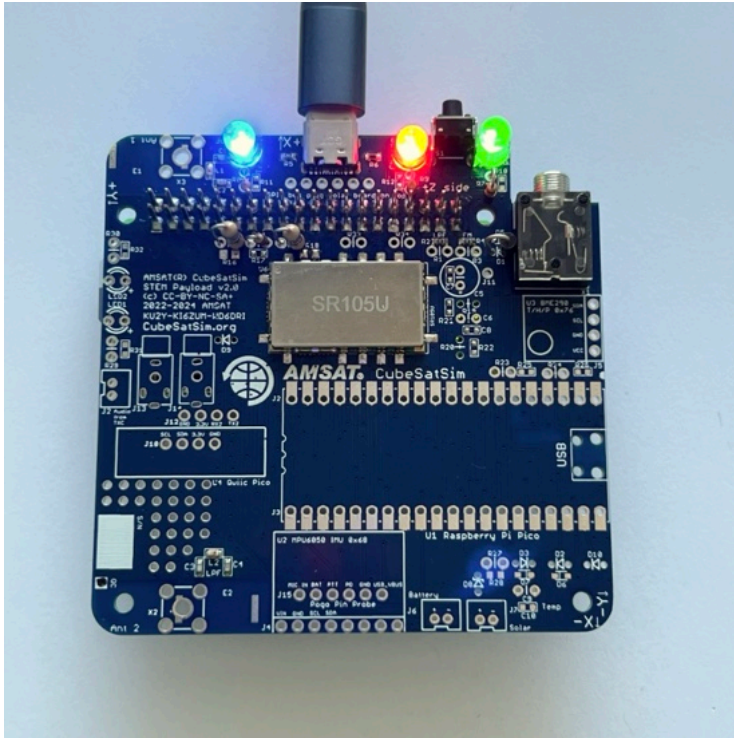
If the LED doesn't illuminate, it might mean there is a problem with the green LED LED3 or resistor R7 or that the CubeSatSim software isn't running.

## Transmit LED Test

---



With the Pi plugged into the main board and the power cable plugged into the Pi micro USB port, after booting, the CubeSatSim software will run. When the CubeSatSim is transmitting, the blue Transmit LED will be illuminated, occasionally blinking. This indicates that the CubeSatSim is transmitting.



If the blue LED doesn't illuminate, it might mean there is a problem with the blue LED LED4 or resistor R8 or the CubeSatSim software. If it only illuminates once, it might mean that the Low-Pass Filter is not detected by the presence of either resistor R1 or R2 - check this resistor.

## Transmit Test

---



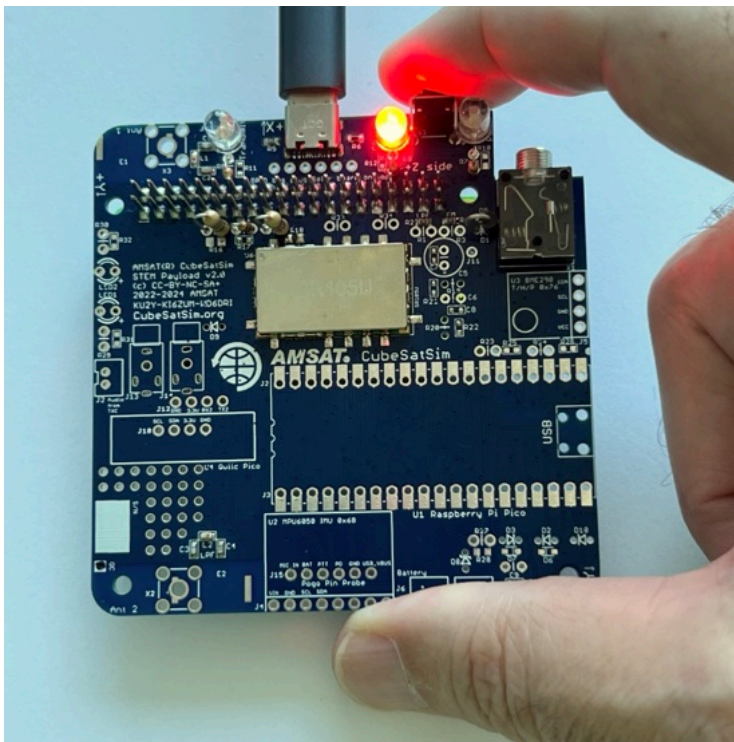
If the blue LED is illuminated or blinking on and off, you can tune your radio or SDR to 434.9 MHz (+/- 15 kHz) to see and hear a carrier signal. You don't have an antenna yet, but a close by radio will pick it up. If you have your FoxTelem ground station working (the [Ground Station instructions are here](#)), you can try to decode frames in FSK mode.

If your blue Transmit LED is illuminated but you don't see a signal, check your antenna on your radio or SDR, and the gain and frequency on your SDR or radio. Or there may be a problem with the CubeSatSim software, in particular the rpitx library.

## Push Button Test

---

Press and hold the pushbutton. The green LED will blink once, then twice, three times, four times, then five times, then it will blink three times slowly. Immediately release the pushbutton when it blinks slowly and the Pi Zero 2 computer will shut down and the transmissions and the blue LED will stop blinking.

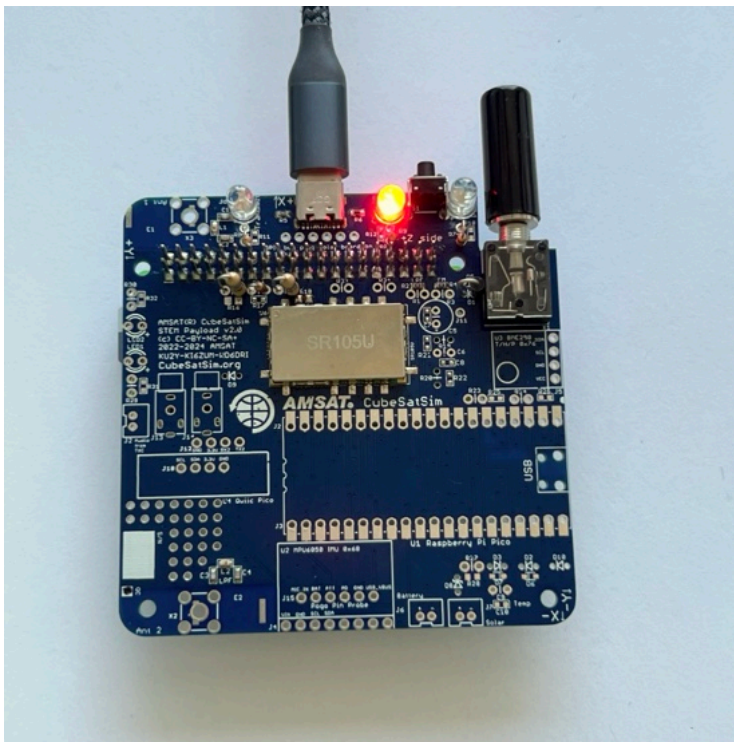


## RBF Switch Test

---

After completing the Push Button Test, insert the RBF pin. Only the red LED will be illuminated.

Unplug the RBF pin and the Pi Zero 2 will boot up and eventually the green LED will turn on the blue LED will start blinking. Hold down the push button again until the Pi Zero 2 shuts down. Then put the RBF pin back in.



## 4.2 Build

---

### Video

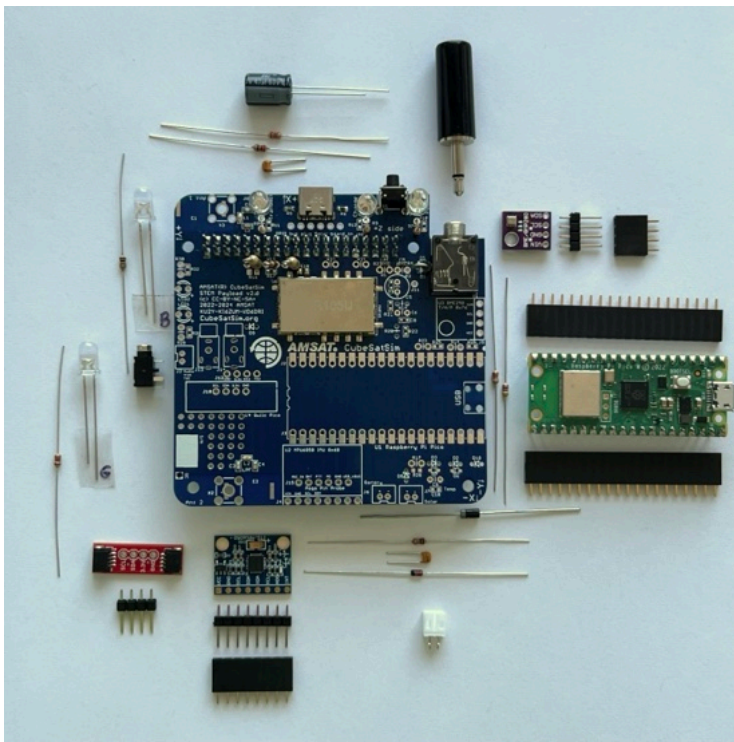
---

Here is a [video of this step](#).

If all your test results are "nominal" (expected results), you can continue your build after shutting down the Pi as described in the Power Button Test (by holding the push button until the green LED blinks slowly then releasing) and unplugging the Pi from the board.

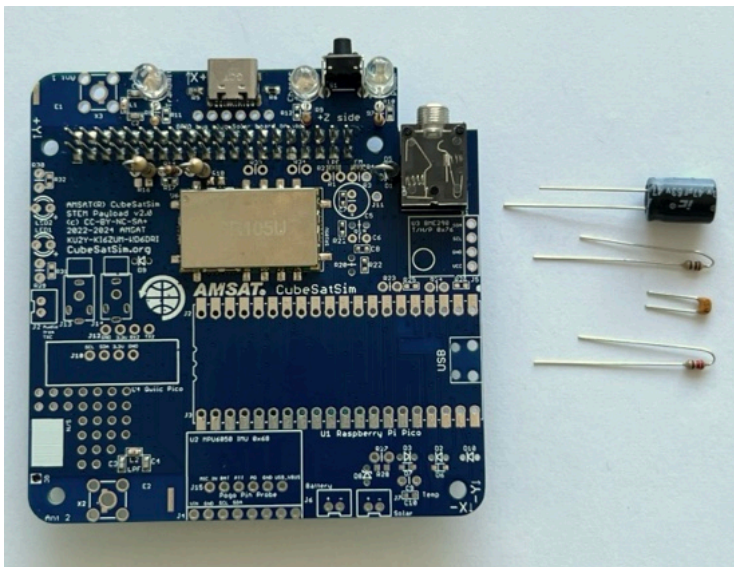
Here are components installed in this step, as described in the BOM

<https://CubeSatSim.org/bom>



You should unplug the Pi Zero 2 from the board before continuing with the assembly.

First, install the audio low pass filter that feeds the PWM (Pulse Width Modulation) audio signals into the FM Transceiver module with 47uF capacitor C5 (can), 100nF capacitor C6 (small yellow capacitor), 220 Ohm resistor R20 (color bands red red brown) and 100 Ohm resistor R19 (color bands brown black brown):

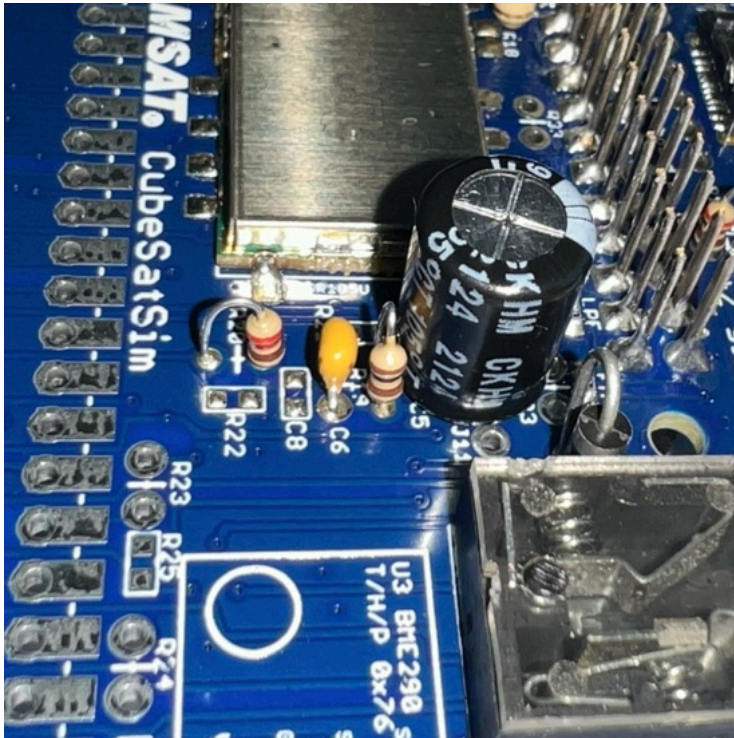




The electrolytic capacitor C5 has polarity - the negative lead (the shorter one in the photo) is marked with a white - . In the photo above and on the PCB, the positive side is marked with a + .

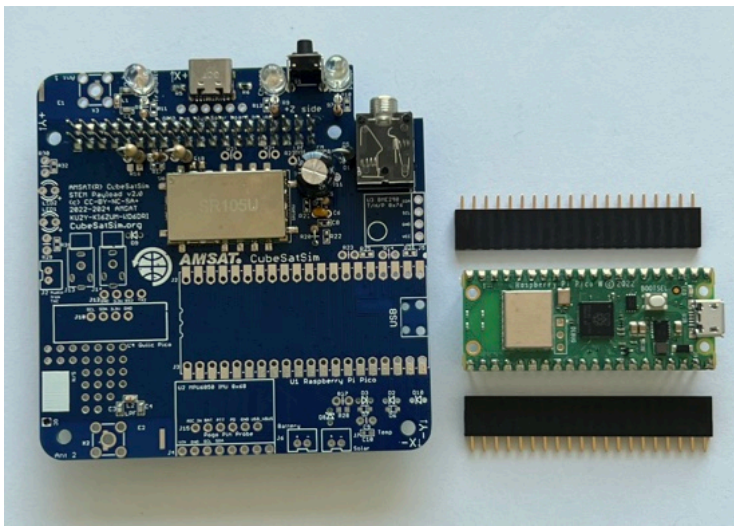


Here is a closeup showing the parts:

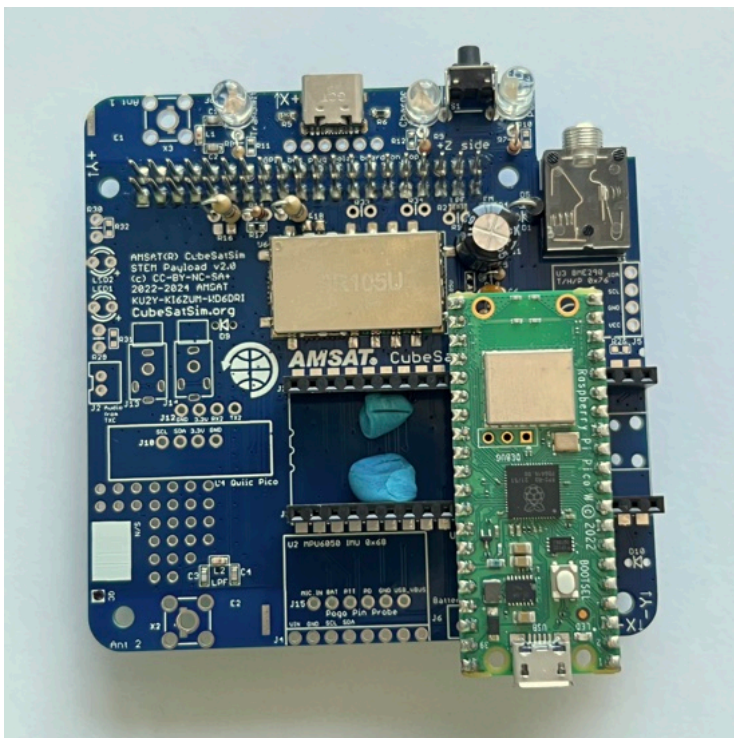


Next, we will install the female sockets J2 and J3 used to mount the Raspberry Pi Pico microcontroller U1. If your Pico doesn't have the male pin headers installed, you will need to solder them in.

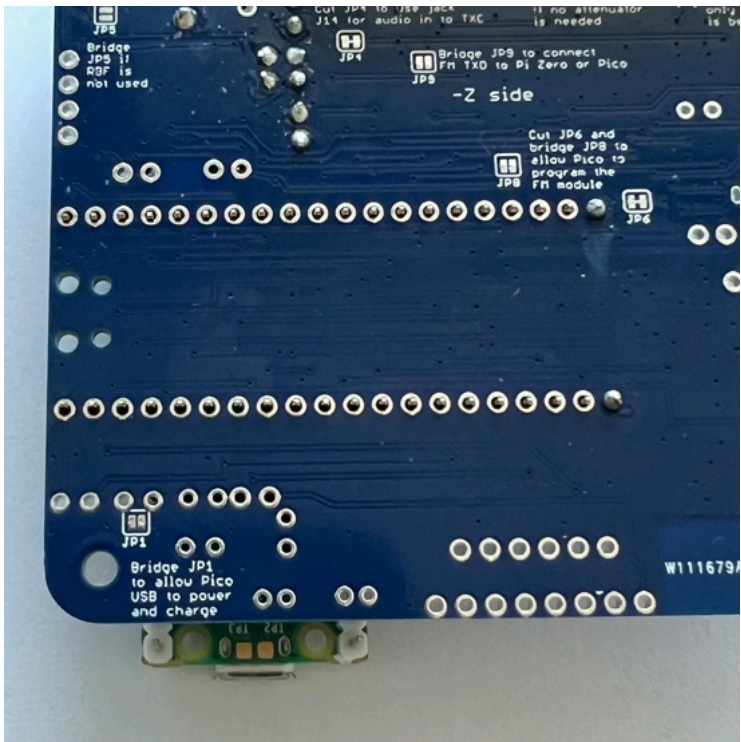




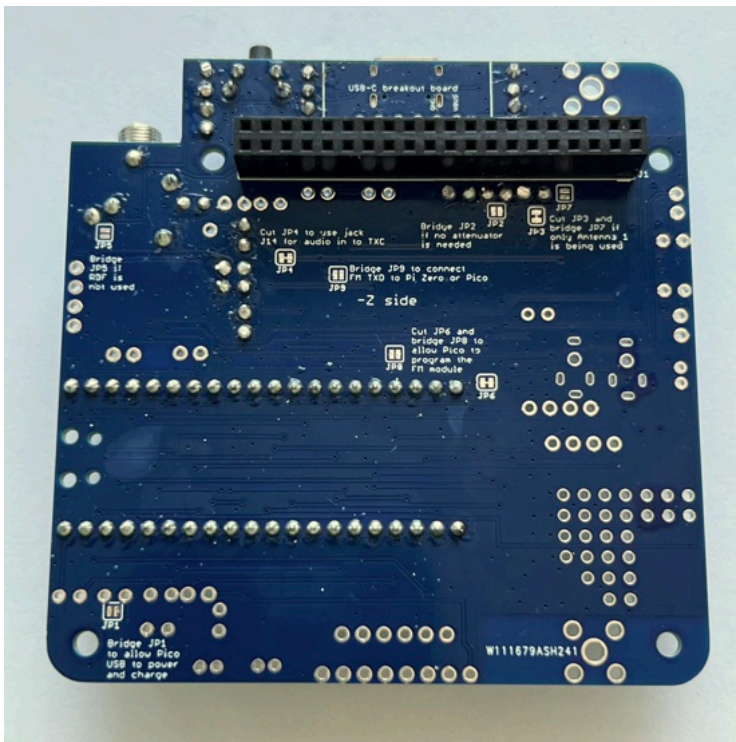
The spacing and alignment of the female sockets is important, otherwise the Pico won't plug in. You can use the Pico itself at right angles to ensure they are spaced correctly and at right angles to the PCB, or you could use the 1x8 pin header from the MPU 6050 board J4.



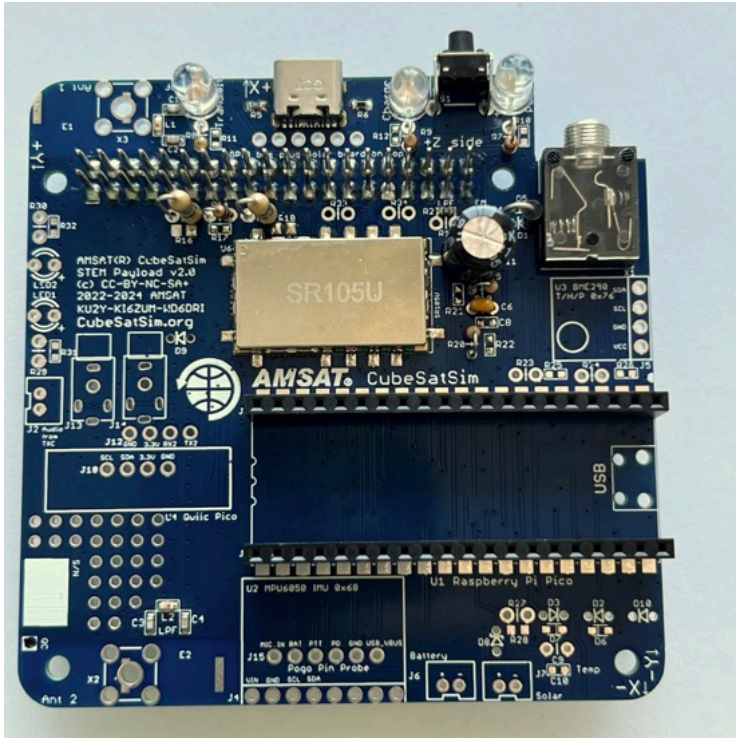
First solder only one pin on each header and double check to make sure it is installed correctly. Look at it sideways to make sure there is no gap - this photo shows a tiny gap which is fine:



Here is the bottom of the PCB with all the pins soldered:

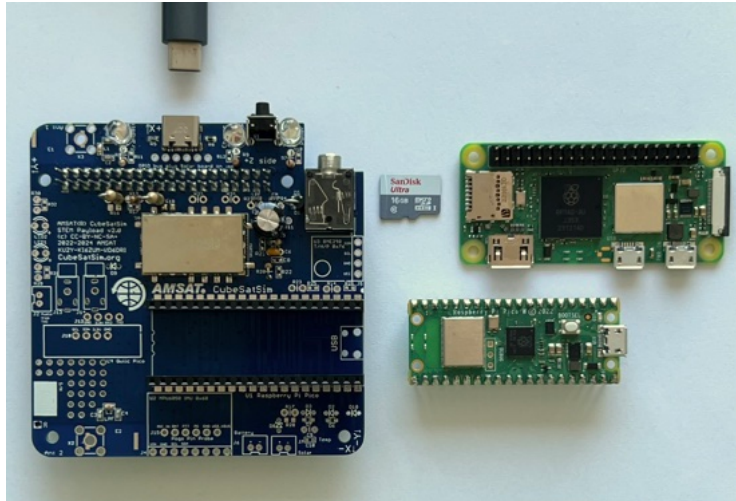


Here is the board with the headers installed ready to plug in the Pico:

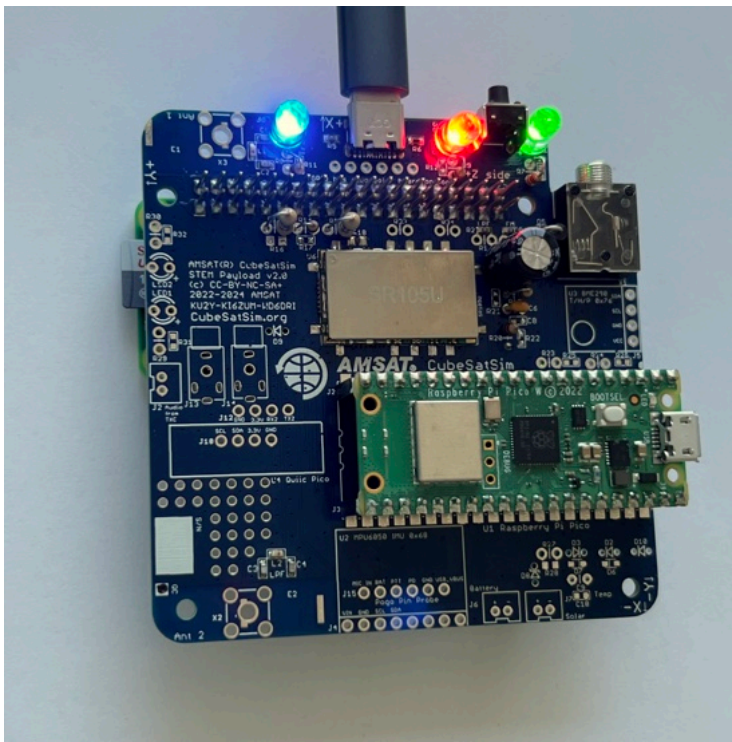




The FM modes can now be tested with the help of the Pi Zero 2 with a programmed SD card and the Pico which has been flashed with the software. If you haven't completed these steps, you can find them here: [Pi Zero 2 and Pico Software Install Instructions](#)



Plug the Pi Zero 2 into the bottom of the PCB and the Pico into the top - make sure the micro USB connector is at the edge of the board as shown below. Plug in the powered USB-C power cable and the Red charging LED should illuminate. The Green Power LED will illuminate after a few seconds. Then the Blue LED will illuminate and go on and off:



If the Pico is programmed and functioning correctly, the small green built-in LED should blink once every second.

## Push Button Test

---

With the Pi plugged into the main board and the power cable plugged into the Pi micro USB port, after booting, the CubeSatSim software will run. When the push button is pressed and held, the green LED should blink rapidly in this sequence: 1 quick blink, 2 quick blinks, 3 quick blinks, 4 quick blinks, 3 slow blinks. Releasing the button after one of the quick blink sequences will put the CubeSatSim in that mode, and the green LED will stop blinking. For example, in this video, the push button is released after the 3 blinks, so the CubeSatSim is in Mode 3.

<http://countingfromzero.net/amsat/v1/blink.MOV>

The modes are: 1: APRS, 2: FSK, 3: BPSK, 4: SSTV, 5: CW.

- When pressed and immediately released,



the Pi should reboot after about 30 seconds.

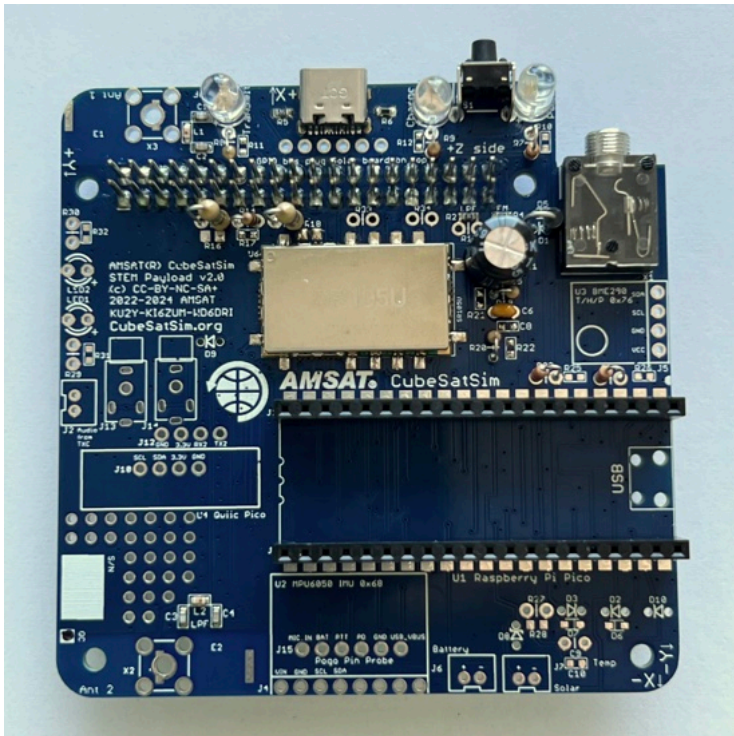
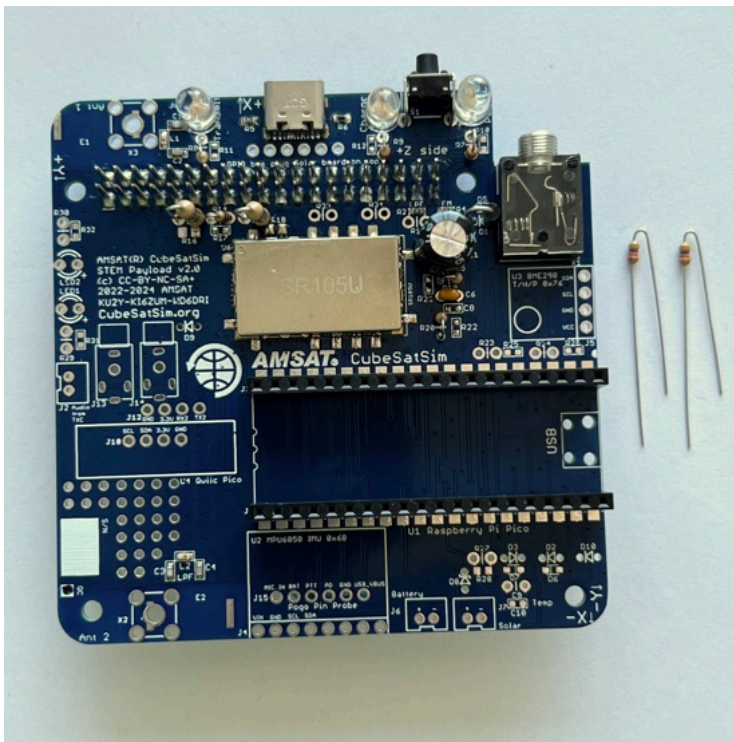
- When pressed and held until the green LED starts to blink slowly (see above for the LED blink sequence), the green LED will then blink slowly three times, then the Pi will shutdown (the green Power and blue Transmit LEDs will go out, then the tiny green LED on the Pi will flash rapidly for about 10 seconds then go out). You can release the button as soon as the LED starts blinking slowly. This video shows the button shutting down the Pi:  
<http://countingfromzero.net/amsat/v1/shutdown.MOV> After the Pi is shutdown (no LEDs are on or blinking), you can safely disconnect the USB power cable and unplug the Pi from the board to continue the build.

If the green Power LED doesn't blink, there might be a problem with the push button switch S1 or with the pi-power-button software.

If you don't hear anything or see anything for the FM modes (APRS, SSTV, and CW), try listening on 450.000 MHz for the signal. If you hear the signal there, it means the FM Transceiver module hasn't had its frequency set to 434.9 MHz by the Pi Zero 2 software. Try powering down everything and then powering up again.

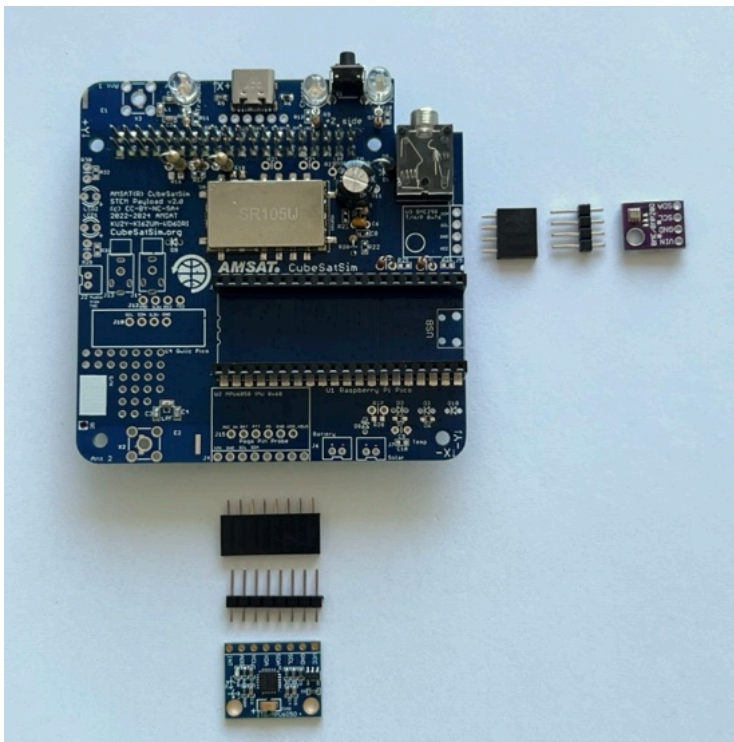
## Continue Build

Then mount the two 4.7k Ohm resistors R23 and R24 (yellow violet red color bands):

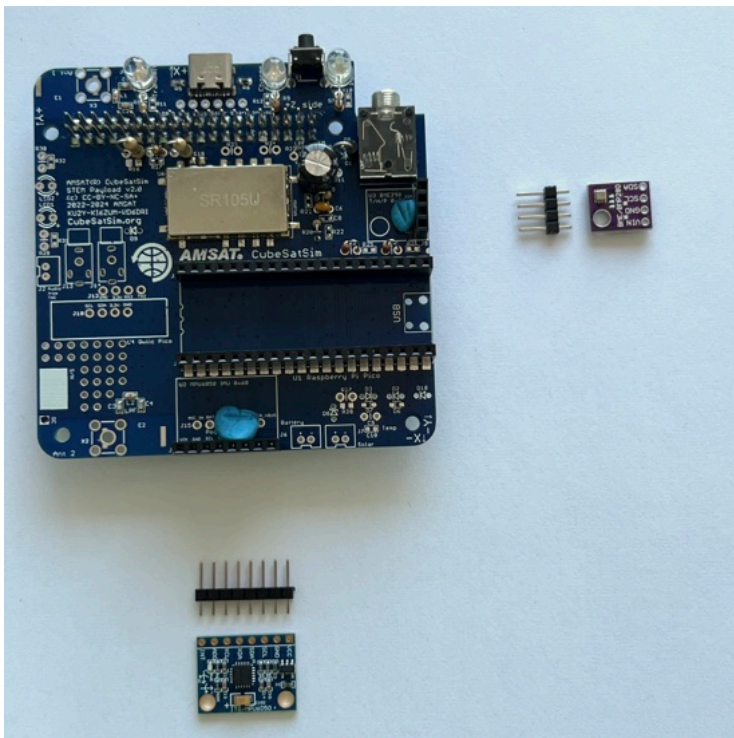


Next we will mount the BME280 sensor (small purple board) J5 and the MPU6050 Gyro and Accelerometer sensor (larger blue board) J4. They can be mounted directly on the board or on female sockets for more flexibility.

They are mounted with female sockets:



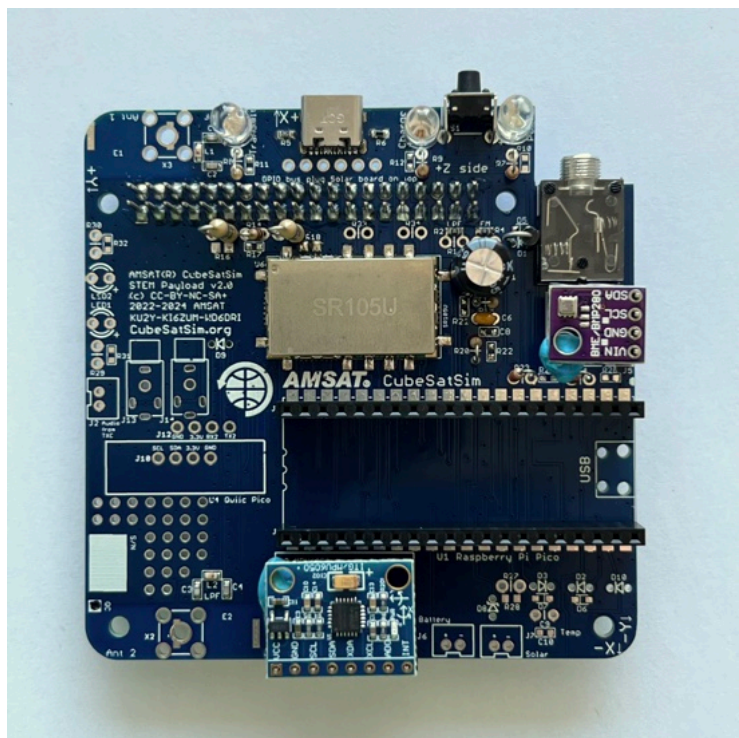
First solder the female sockets, making sure they aren't tilted. Blue putty can be used to hold them in place during soldering.



Solder one pin on each socket, double check, then solder the rest.

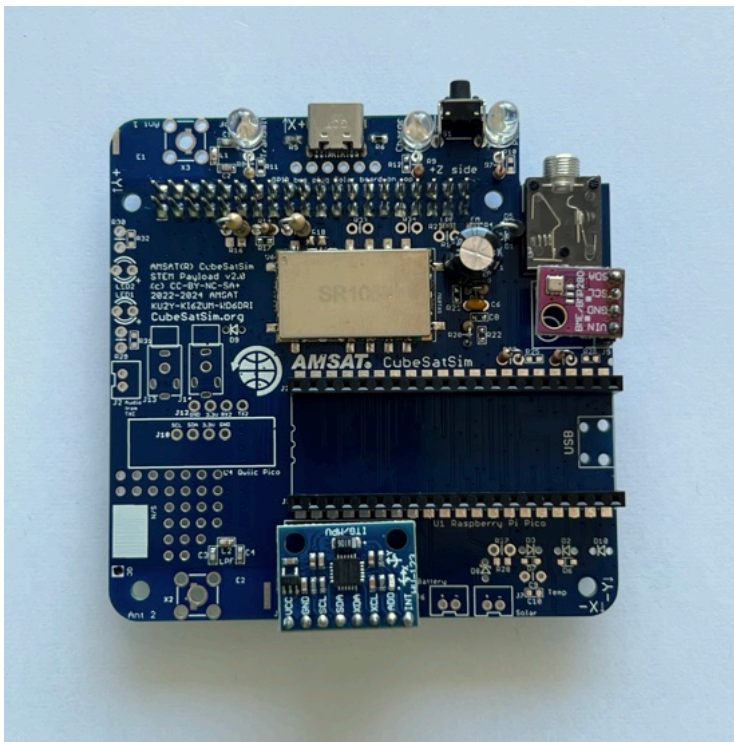


Insert the pin headers into the sockets, long pins into the sockets, then place the sensor boards on top. Blue putty can be used to hold them horizontal to the board.



**IMPORTANT:** make sure the purple BME280 is not upside down. The top has the BME sensor chip (small metal can). The circular hole in the board should be to the left.

Solder the pins:

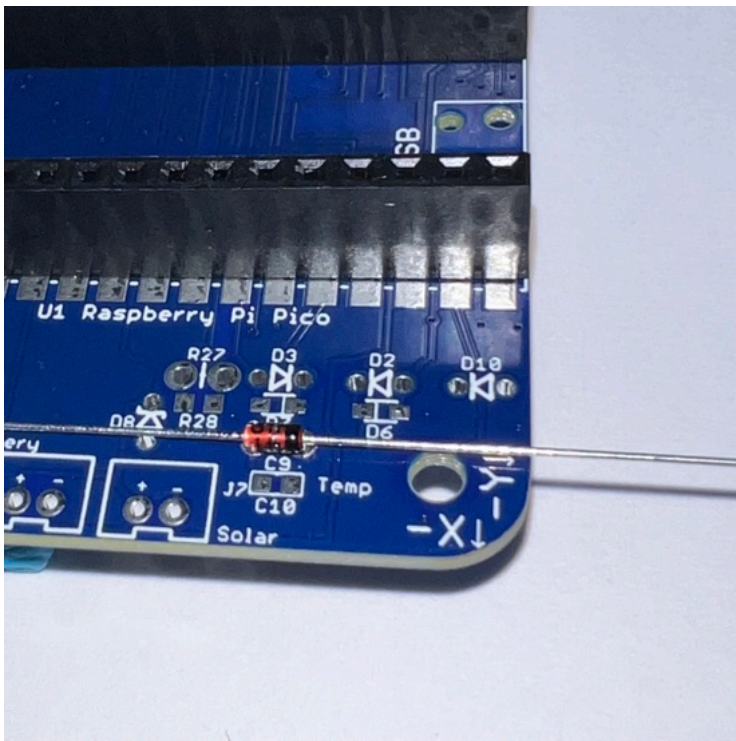
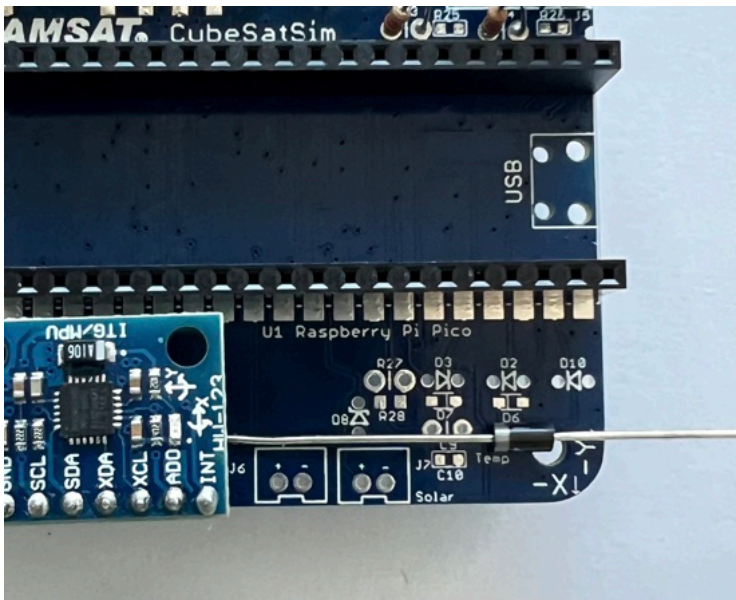


Next, solder these four components on the bottom right of the PCB:

- 10k resistor (black brown orange color bands) R27
- 100nF capacitor C9 (small yellow capacitor)
- 1N5817 diode D2 (black device with a white band on one side to mark polarity)
- 1N4148 diode D3 (has a glass case with the numbers "41" and "48" written on it and a black band to indicate polarity)

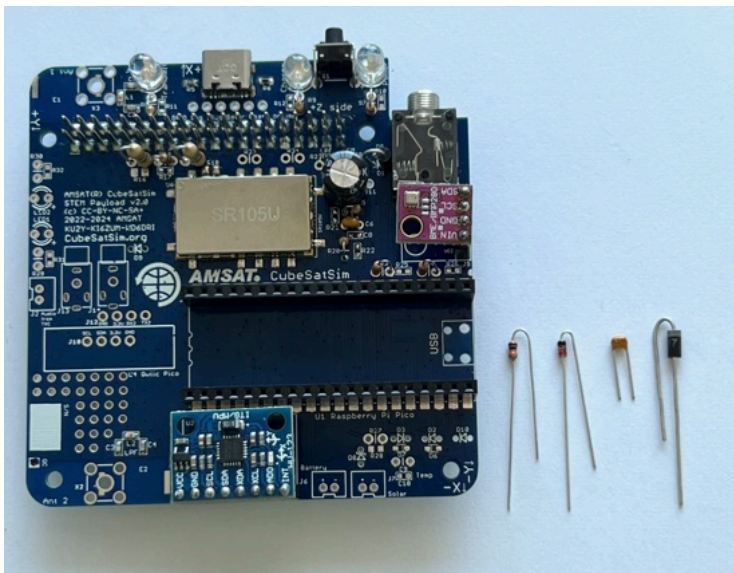
Here's the polarity of the diodes:





All the diodes, along with all the resistors on the board are mounted vertically. Bend over one lead on the diode and insert, making sure the polarity of the diodes is correct before soldering them.

Here's all four parts ready to install:



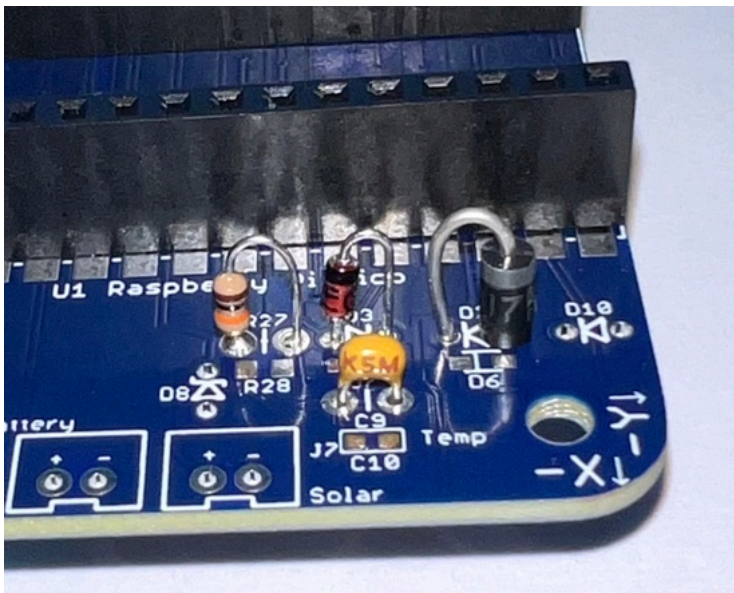
Solder the leads on the bottom of the PCB.

Then trim the leads, being careful to wear safety glasses and putting your finger over the end so it doesn't go flying.

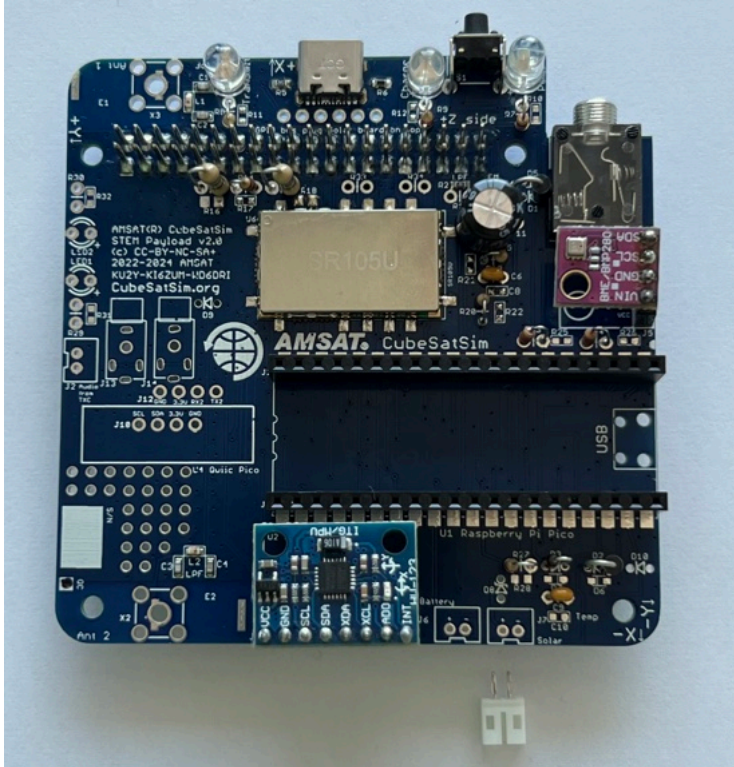
Here's after they are installed:



Here is a closeup:

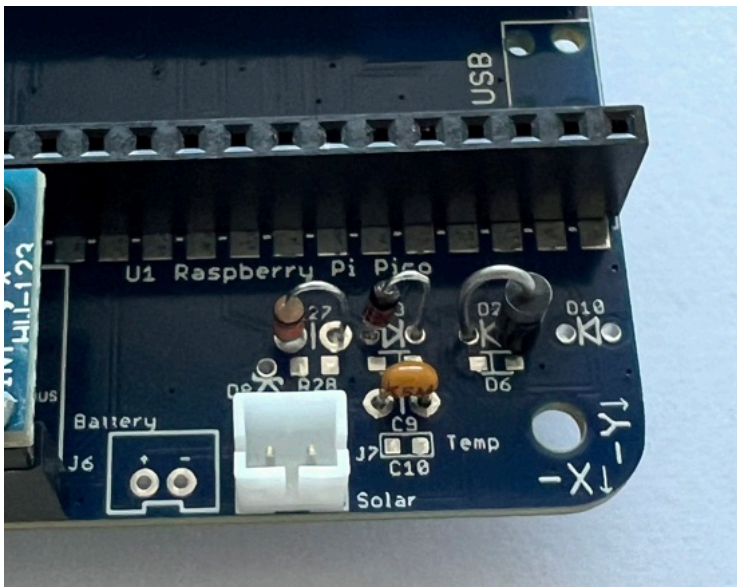


Install the JST connector J7 which is labeled "Solar" since the JST jumper cable will connect it to the Solar Board. The other JST connector J6 labeled "Battery" is used if no solar panels are used and the Battery Board will be connected here.

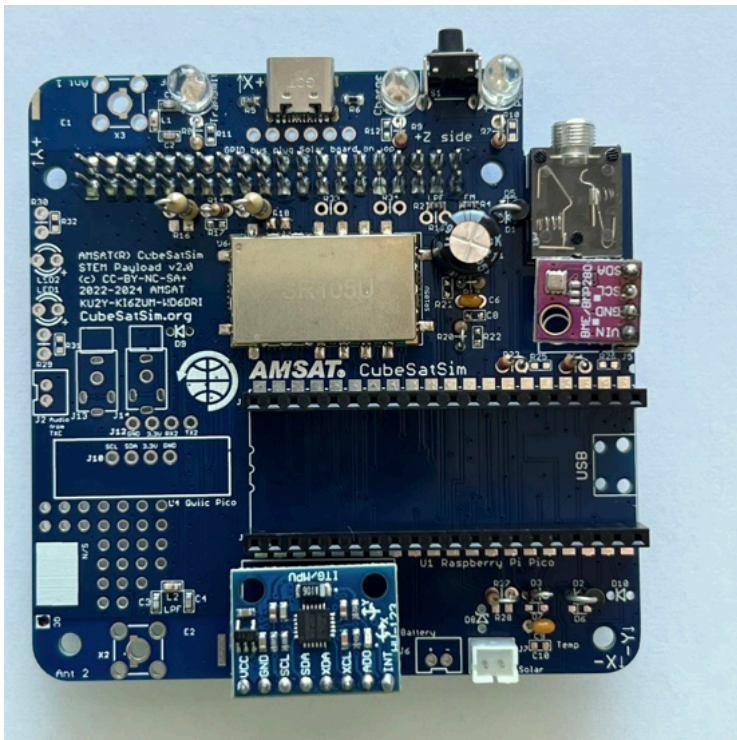


Here is a closeup showing that the slot should face the edge of the PCB for correct polarity. Some JST connectors snap into the board. If yours do not, then I recommend a drop of super glue gel or other adhesive prior to soldering:





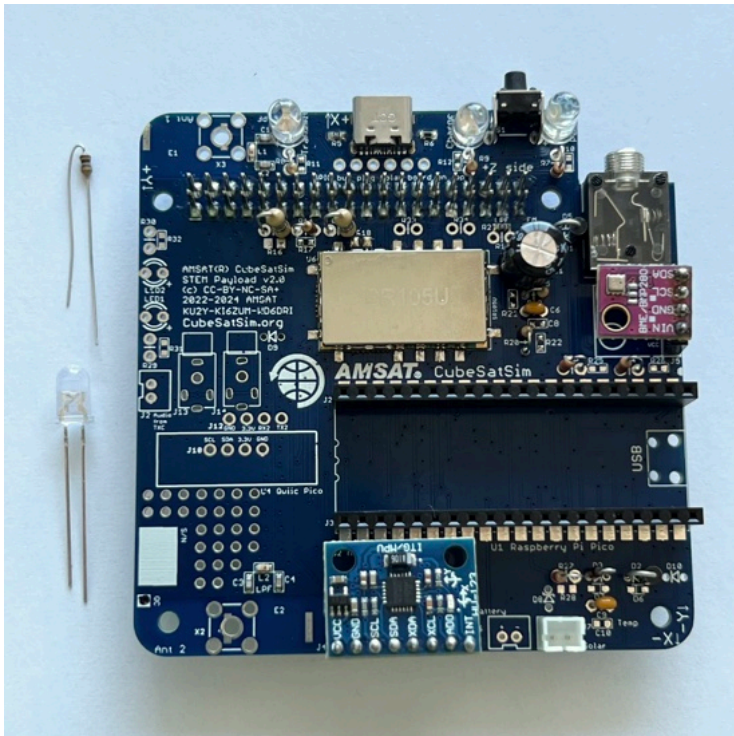
Here's how it looks installed:



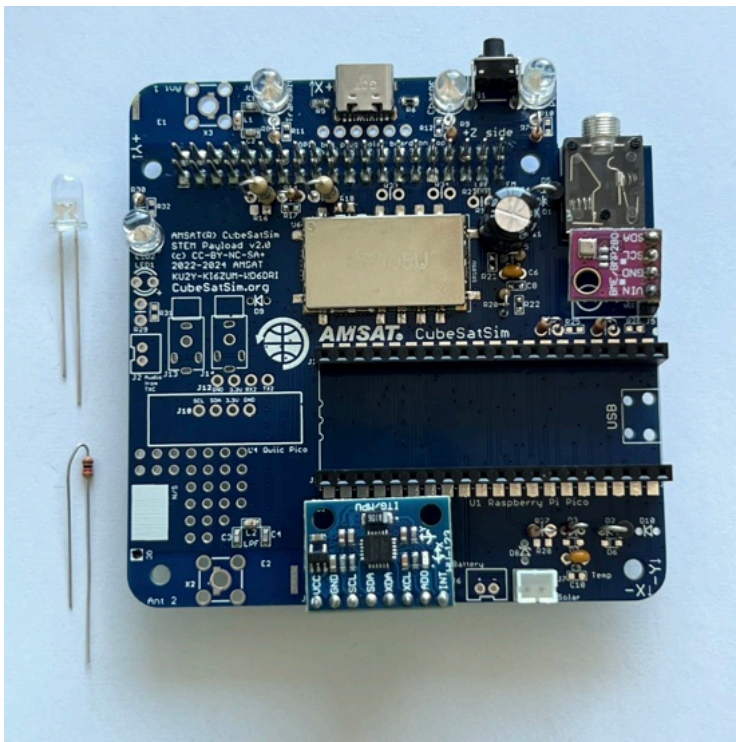


Then install the blue LED2 and 100 Ohm R30 (brown black brown color bands). LEDs need to be installed with the correct polarity (one lead is + polarity and the other lead is - polarity) or it will not illuminate. The longer leg on the LED is the '+' lead and should be away from the edge of the PCB. Also, if you look at the LED lens from the top, it is circular but there is a flat side that marks the '-' side. So the flat side of the LED should be towards the edge of the PCB.

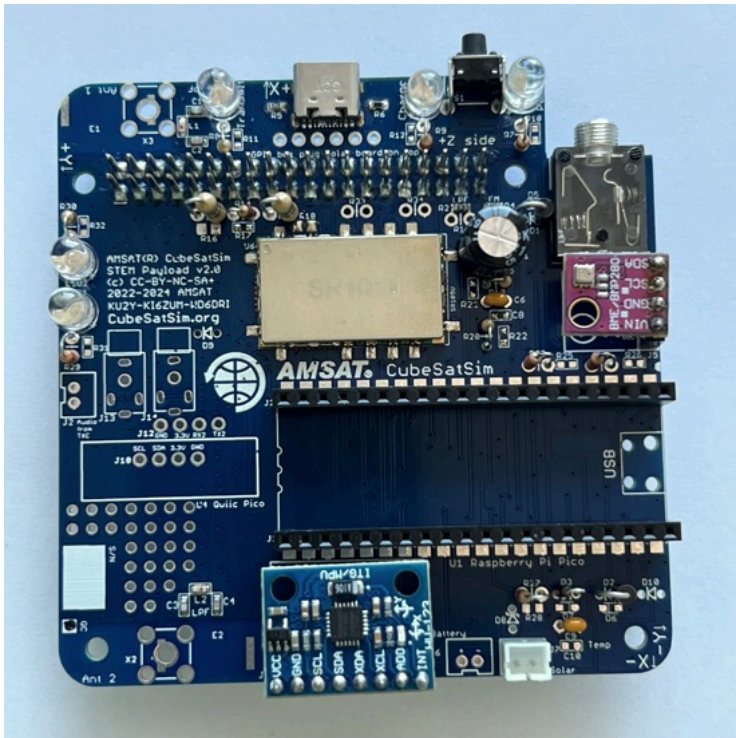
The longer lead of the LED is the positive and matches the '+' on the PCB. This LED and the green one next to it are turned on and off by the microcontroller.



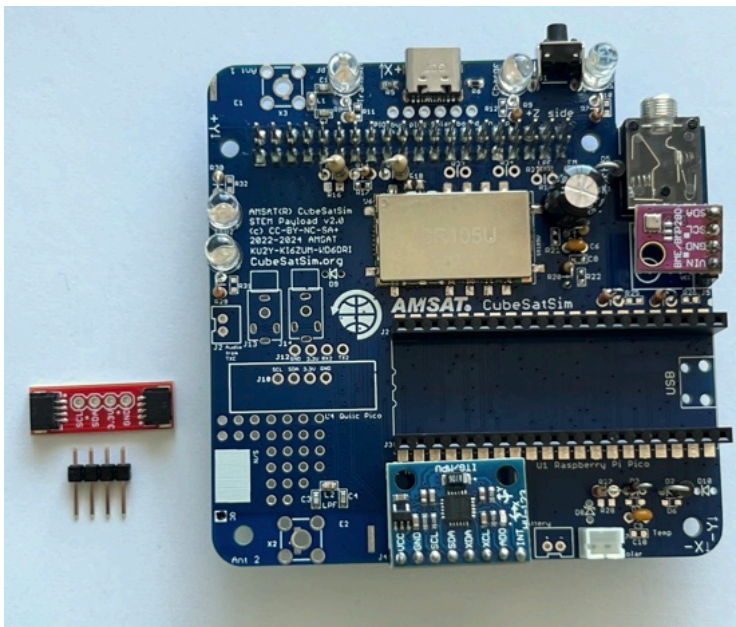
Then install the green LED1 and 1k Ohm R29 (brown black red color bands). The longer lead of the LED is the positive and matches the '+' on the PCB.



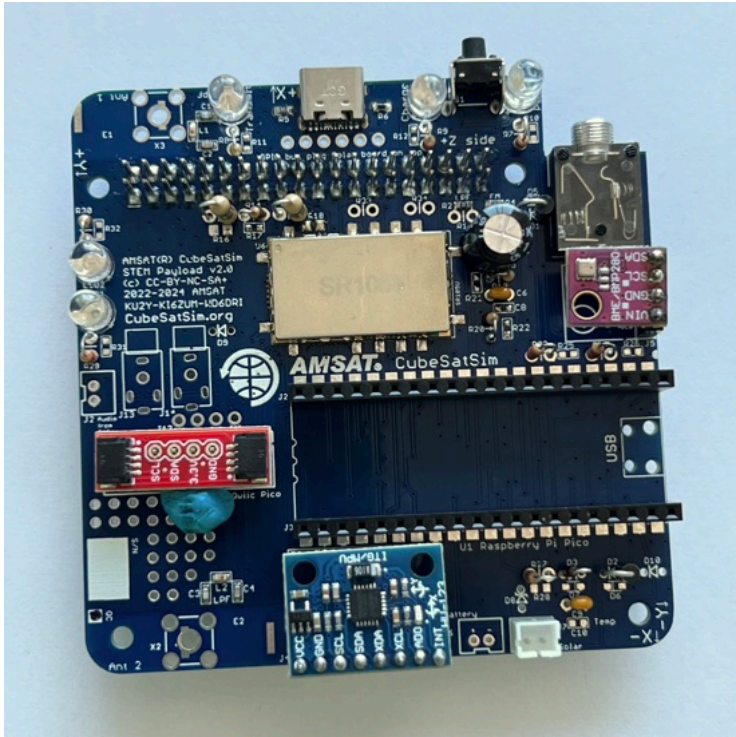
Here are all the resistors and LEDs installed:



Next, install the Qwiic connector (red board) and 1x4 male pin header J10 for adding more I2C sensors to the board:

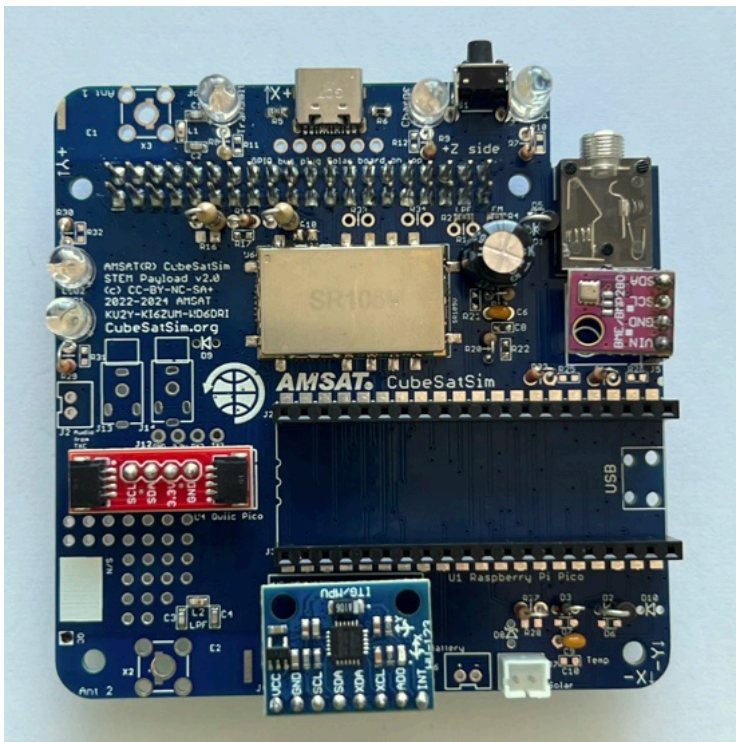


First install and solder the four pin male header J10, long pins down:

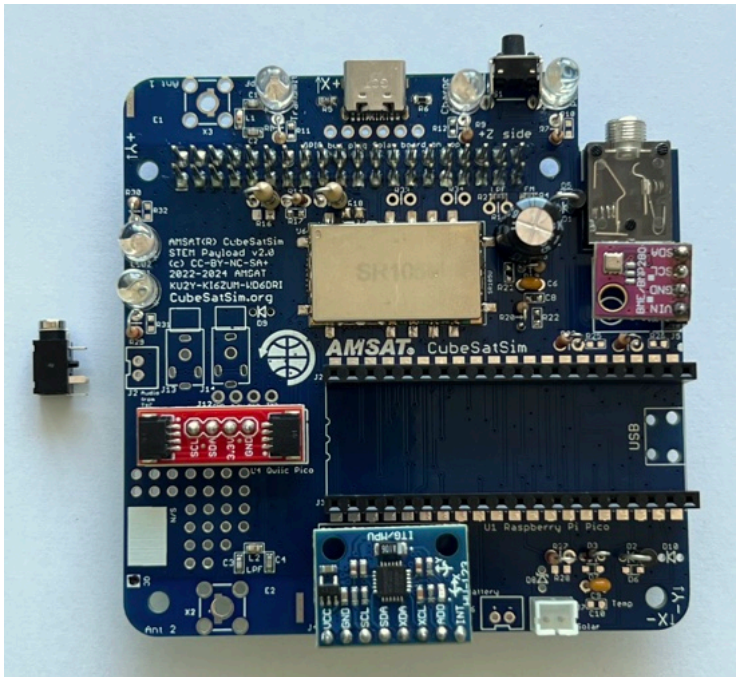


Then, install the connector, using blue putty to hold it level:





Next, install the 2.5mm jack J13.



Solder it in place:

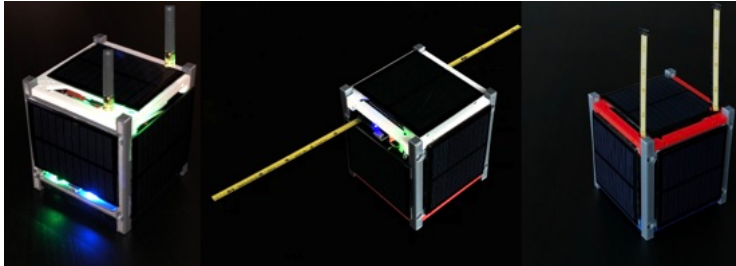




## 4.3 Antenna Install

---

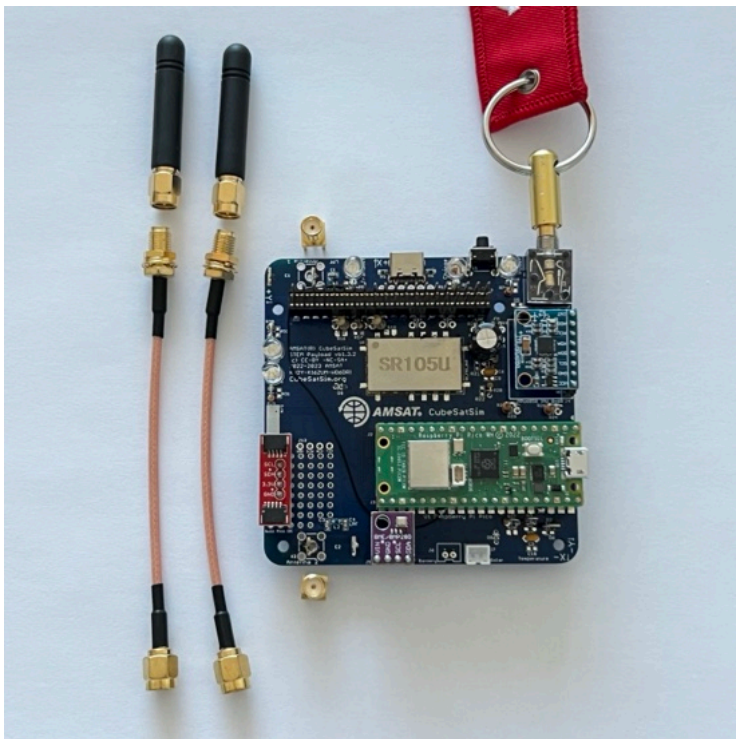
You have a choice of antenna. You can build a SMA antenna (left), a tape measure dipole (center), or two tape measure monopoles (right) as shown here:



### SMA Antennas

---

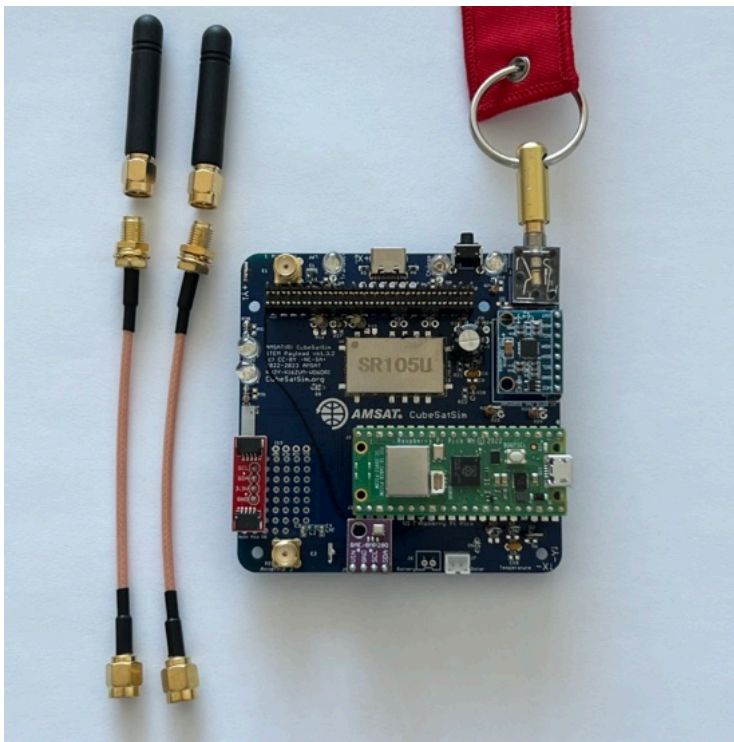
To connect antennas using SMA connectors, you just need to solder two SMA connectors onto the Main board.



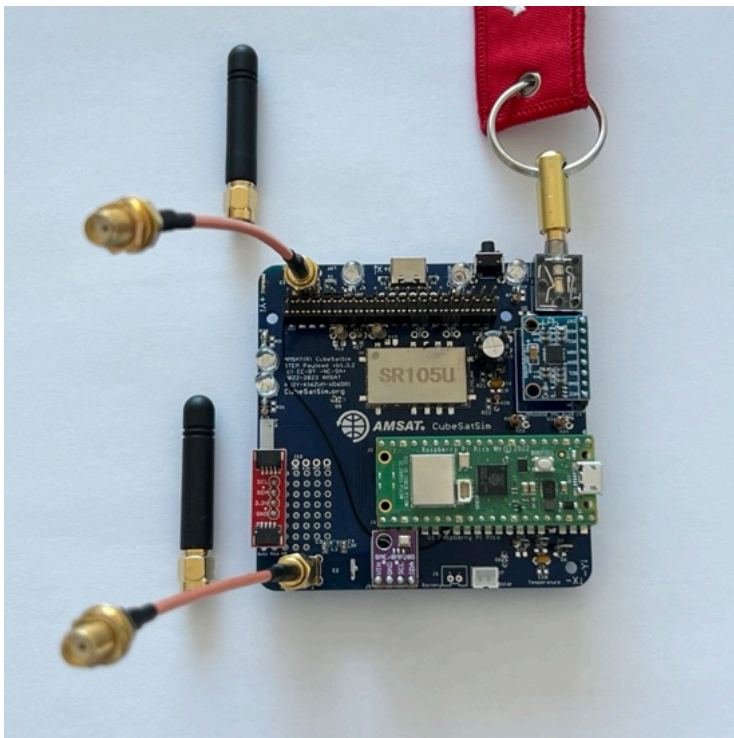
This part takes a lot of heat, so make sure your iron is set to high heat. You might also want to apply liquid flux to the pins to help solder it.







You will connect SMA coax cable and a rubber duck antenna to each SMA connector when you put the frame together.



## Tape Measure Dipole Antenna

---



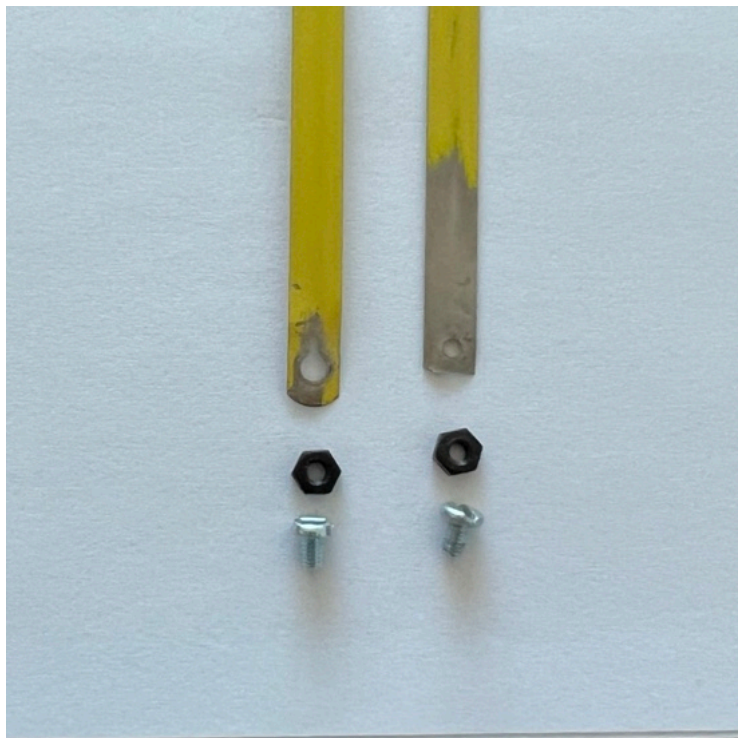
Alternatively, you can make a tape measure dipole or monopole (vertical) using these parts:

- 1/4" Tape measure (3/8" and larger ones work too)
- Two metal M2.5 screws (nylon will not make electrical contact)
- Two M2.5 nuts, nylon or metal since it is just a spacer to keep the tape measure from touching the PCB.

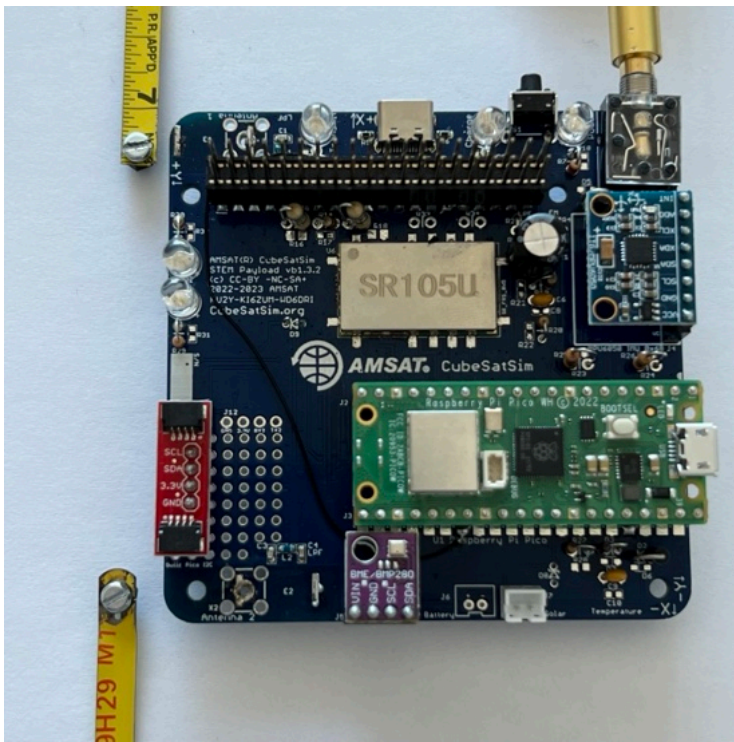
Extend all the tape out of the measure before cutting or the remaining tape will retract inside and you will need to break open the case. Be careful of sharp edges, and put electrical tape over the cut edges immediately.



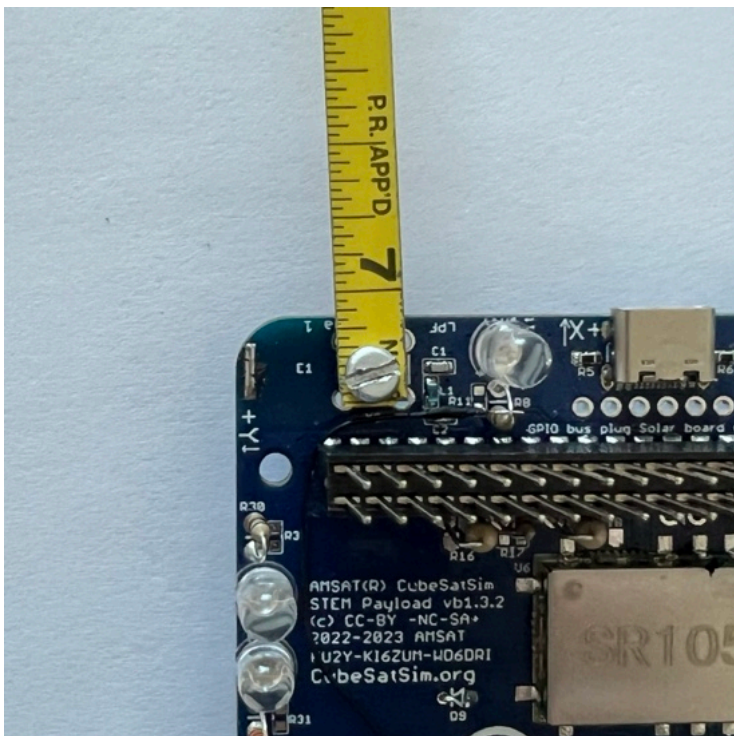
For the dipole antenna, you will need two  $1/4$  wavelength lengths of tape measure, each with a hole drilled in it. Drilling the hole can be tricky, so drill the hole first, then cut the tape to length. Also, using a punch or small screwdriver to dent the tape in the middle can help prevent the drill from moving around. Also, start with the smallest drill bit you have, ending up with a  $3/32$ " drill bit. It doesn't matter if the hole is slightly off center. Using sand paper or emery cloth, remove the paint from the tape around the hole on the bottom (no numbers) side of the tape measure. Cut to length after you have successfully made the holes, approximately 6.5 inches:



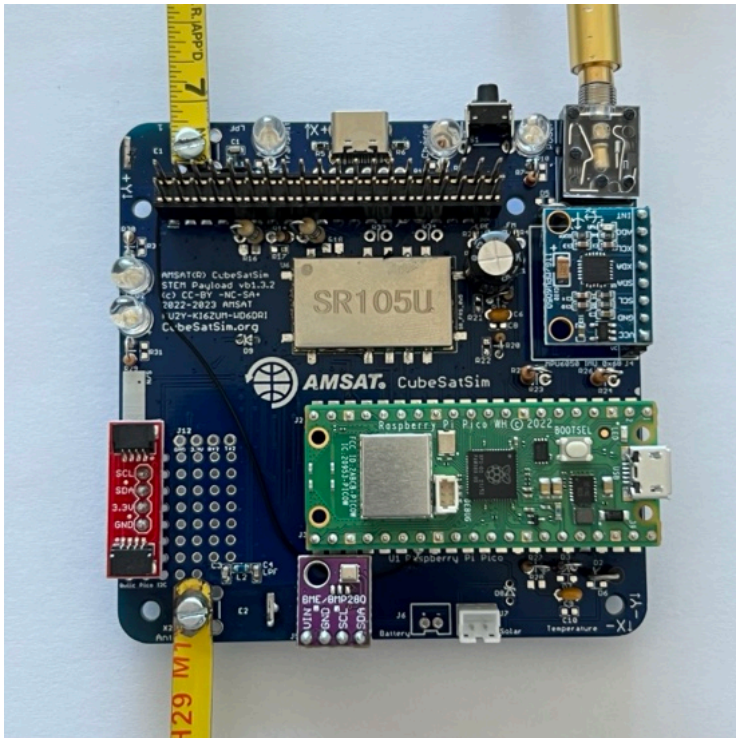
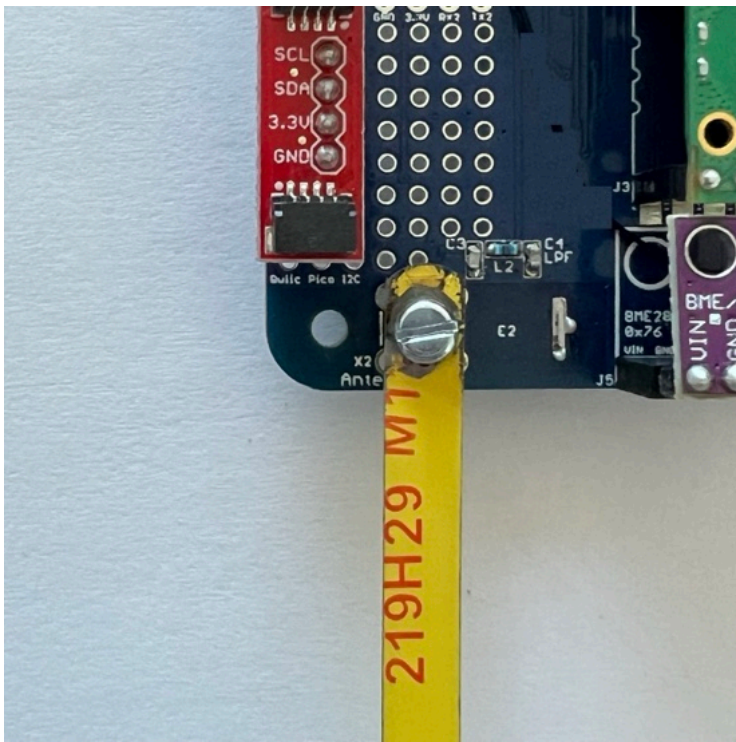
Put the screw through the hole in one of the tape lengths. Then screw the nut onto the screw but don't tighten it all the way.



Press down with the screwdriver as you screw the screw into the PCB and the screw will cut threads into the center hole of the antenna connector as shown. It can be easier to start the screw in the hole without the tape measure at first to begin cutting the threads.

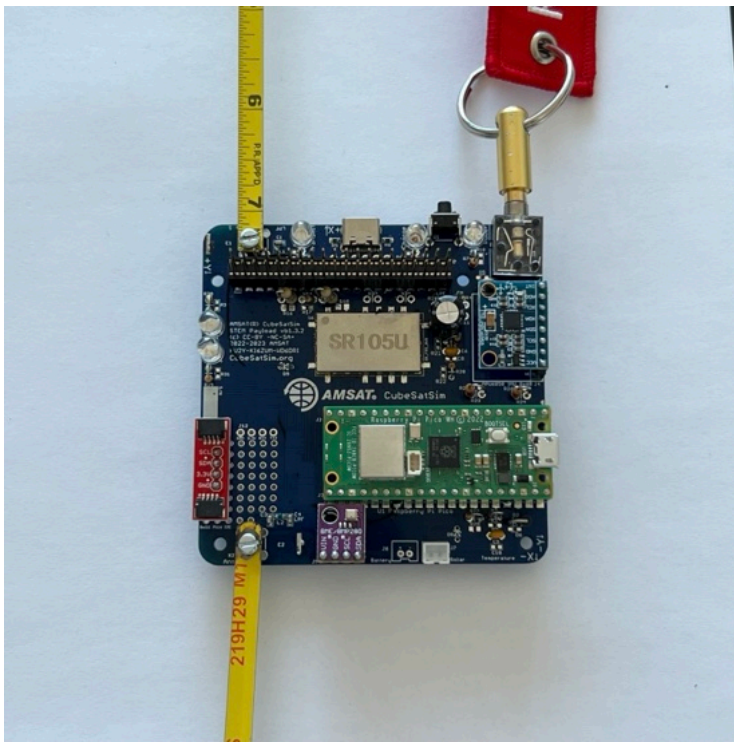






The two tape measure antennas will go through the gap between the solar panels in the frame.

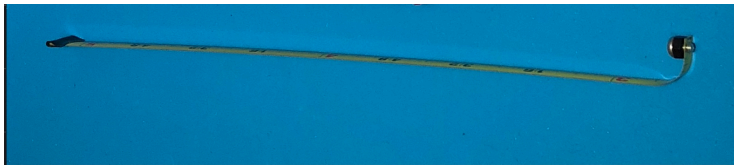




## Tape Measure Monopole Antenna

---

To build the monopole, you need to cut two lengths of tape measure to the same length as the dipole, approximately 6.5 inches long. You need to bend the tape measure next to the hole to a right angle, but don't fold it as it will break off



Screw the screw into the board so the monopole antenna goes vertical. There will be a slot in the top frame piece for the tape measure antennas to stick out.

## 4.4 Testing

---

Video

Here is a [video of testing the Main Board](#).

## Payload Sensor Testing

You can now test that the Raspberry Pi Pico can communicate with the two sensor boards. Your Raspberry Pi Pico should already have the software installed from Step 2. If not, go back to Step 2 and install it.

You will connect a micro USB to the Pico and to your computer which has the Arduino IDE installed.

(TBD photo showing Main Board powered through the micro USB cable connected to the Pico)

In the Arduino software, open the Serial Monitor by clicking on the magnifier icon in the top right of the window. In the Serial Monitor window, make sure 115200 baud is set in the lower right. You should see a response displayed similar to this:

```
OK BME280 24.89 1003.96 77.61 21.13
MPU6050 -0.85 -2.48 -1.09 0.19 -0.15
1.02 XS 0 0 0.00
```

The `OK` is the status response. The four numbers after the `BME280` are the temperature in Celsius, pressure in hPa, altitude in meters, and humidity in percentage read from the purple BME280 sensor. The six numbers after the `MPU6050` are the X, Y, and Z axis angular rotation in degrees per second and the X, Y, and Z axis acceleration in g. If you get a series of zeros after a sensor, it means it was not successfully read by the Pro Micro. The three numbers after `XS` are extension sensor fields that you can set in the code by setting the `Sensor1`, `Sensor2`, and `Sensor 3` variables.

If you get all zeros for a sensor, you need to determine if it is a hardware or software problem. Running an I2C bus scanner program will tell you if the sensor can be accessed on the I2C bus.

The blue MPU6050 should be at address `0x68` while the purple BME280 should be at address `0x76`. If neither device is present on the I2C bus, make sure resistors R1 and R2 are soldered in and are 4.7k in value. If one sensor shows up but not the other, it might be due to soldering on the sensor pins or due to a bad sensor board.

If both sensors show up on the I2C bus but you get zeros in the Serial Monitor, this indicates a software problem.

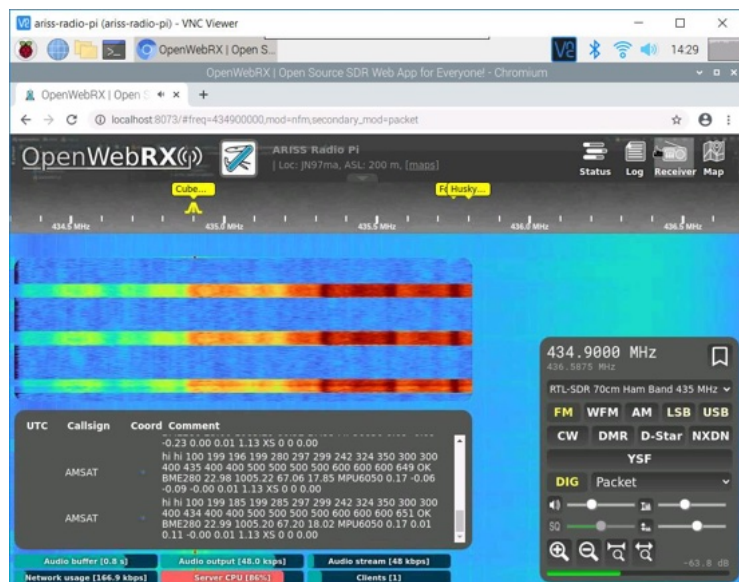
## **Complete Payload Test**

When the Pi Zero 2 is plugged into the Main board, the Pi will read the sensor data over the UART and report the data in telemetry. Connect the USB-C power cable into the Main Board and remove the RBF plug to power it up.

(TBD photo showing Pi Zero 2 plugged into the bottom of the Main board powered through the USB-C cable)

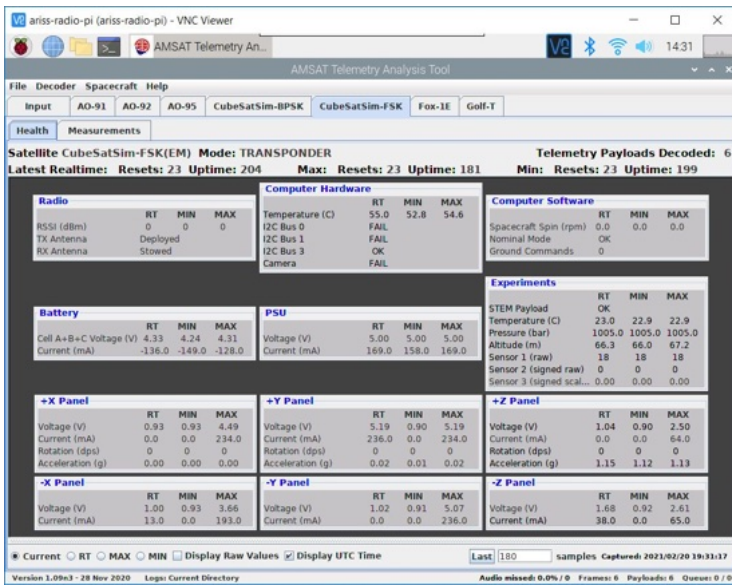
When the CubeSatSim software is running, you should see the built-in LED on the Pico blink once, then twice, then repeat.

In APRS (AFSK 1200) mode, the sensor string will be added to the end of the data.

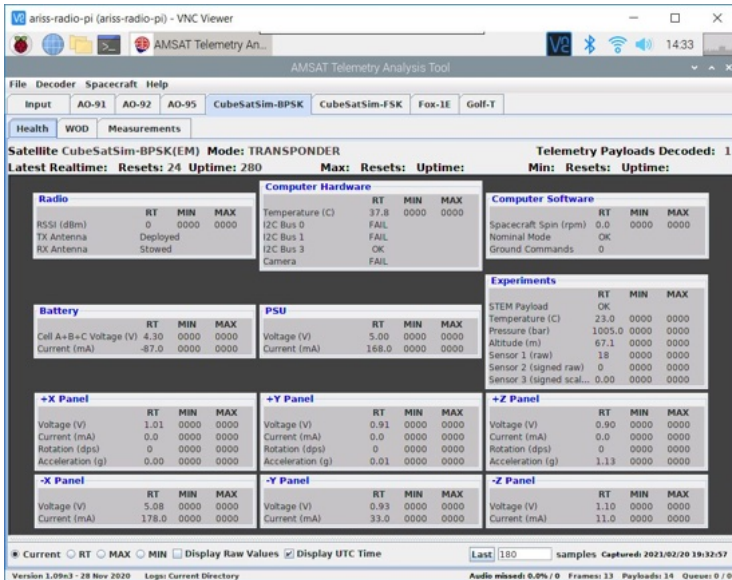


In FoxTelem, in DUV/FSK mode, under the CubeSatSim-FSK tab Health tab, Experiment box, you should see STEM Payload OK and numbers for the other fields. In addition, the rotation and acceleration will also be displayed under the +X, +Y, and +Z boxes.





Using the pushbutton, you can switch the CubeSatSim to BPSK mode by pressing and holding the push button until it blinks rapidly three times. Then, if you run FoxTelem in BPSK mode, you can look under the CubeSatSim-BPSK tab and see:



On the spinning turntable, you should see 7-10 degrees per second of rotation and the blue LED on the STEM Payload board illuminated. If the CubeSatSim is accelerated greater than 1.2g, the green LED will illuminate on the STEM Payload board.

# Payload Serial Connection Troubleshooting

---

If you see the right output in the Serial Monitor in the Arduino IDE, but you don't see the payload data in the APRS packet or FoxTelem says STEM Payload FAIL, here are some things you can check.

Login to your Pi and type these commands:

```
cd
```

```
CubeSatSim/log | grep ayload
```

The use of `ayload` instead of `payload` will give you the log entries for `Payload` and `payload`.

You can also send commands over the serial port like you did with the Arduino IDE using these commands:

```
sudo apt-get install -y minicom
```

```
sudo systemctl stop cubesatsim
```

```
minicom -b 115200 -o -D /dev/serial0
```

Now you should see the same output you saw in the Arduino Serial Monitor.

To exit, you have to type `Control-A` then `z` then `x` then hit Return.

## Adding Additional Sensors

---

Here's how to [add more sensors to your Main board](#).

## Additional Sensor Info

---

The BME-280 Temperature Humidity Barometric Pressure Sensor (small purple board):

<https://lastminuteengineers.com/bme280-arduino-tutorial/>

Here is some code that displays all the values:

<https://github.com/alanbjohnston/CubeSatSim/tree/master/stempayload/bme280test>

The MPU-6050 (GY-521) 3-Axis Accelerometer and Gyro (larger blue board with green LED):

<https://howtomechatronics.com/tutorials/arduino/arduino-and-mpu6050-accelerometer-and-gyroscope-tutorial/>

Here is some code that displays all the values:

<https://github.com/alanbjohnston/CubeSatSim/tree/master/stempayload/GetAllData>

The next step is to [build the Battery board](#).

+ Add a custom footer

# 5. Battery Board

[Edit](#) [New page](#)

Alan Johnston edited this page 11 hours ago · [24 revisions](#)

If the images fail to load, you can [download a PDF of the page](#).

## 5. Battery Board

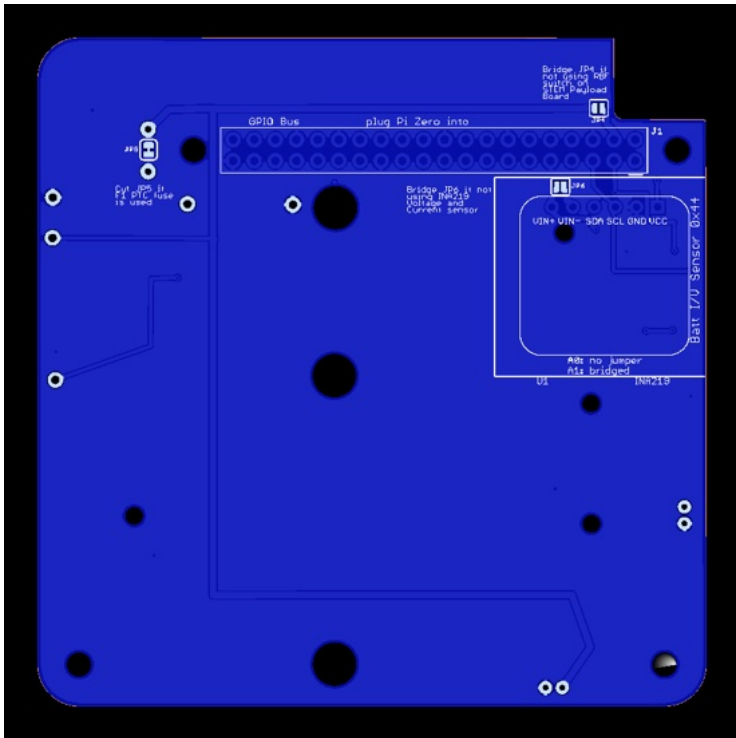
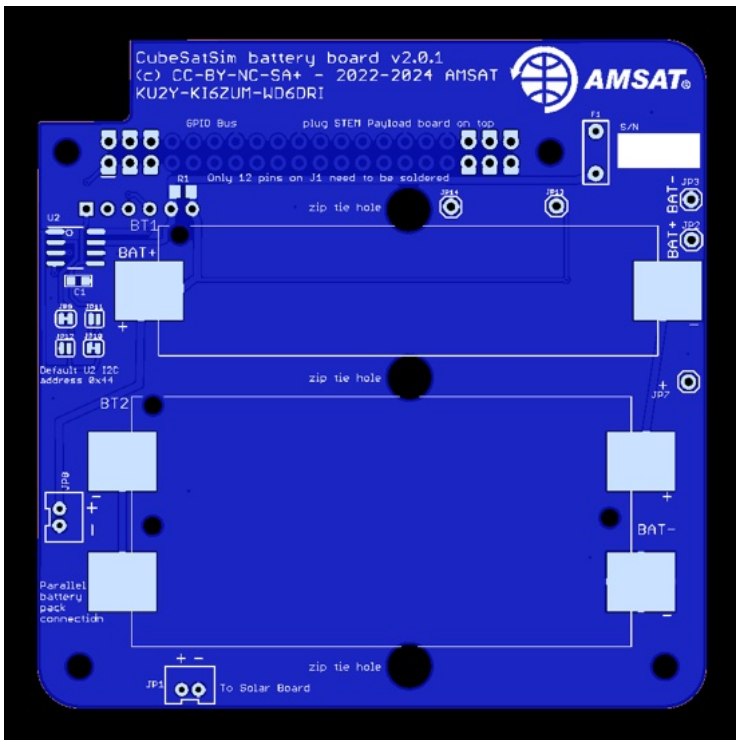
These instructions are to build and test the Battery board version v2.0.



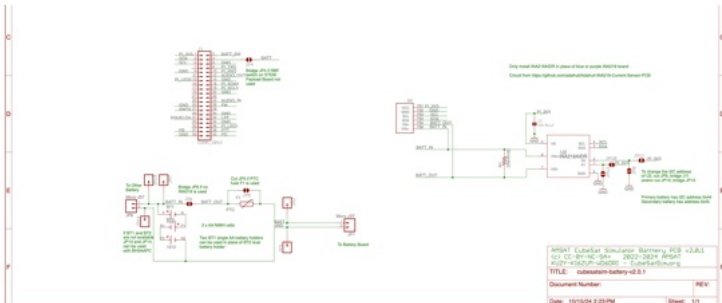
Here is the PCB top and bottom:

- Pages 147
- Find a page...
- Home
- 1. Main Board 1
- 2. Software Install
- 3. Ground Station
- 4. Main Board 2
- 5. Battery Board
  - 5. Battery Board
  - Checklist
  - Battery Board Instructions
  - Video
  - Assembly
- 6. Solar Board
- 7. Solar Panels and Frame
- 8. Board Stack
- 9. Final Testing
- Adding New Sensors
- Command and Control





Here is the schematic:



- ▶ [Creating the CubeSatSim Raspber...](#)
  - ▶ [CubeSatSim Lite](#)
  - ▶ [CubeSatSim Loaner User Guide](#)
- Show 132 more pages...

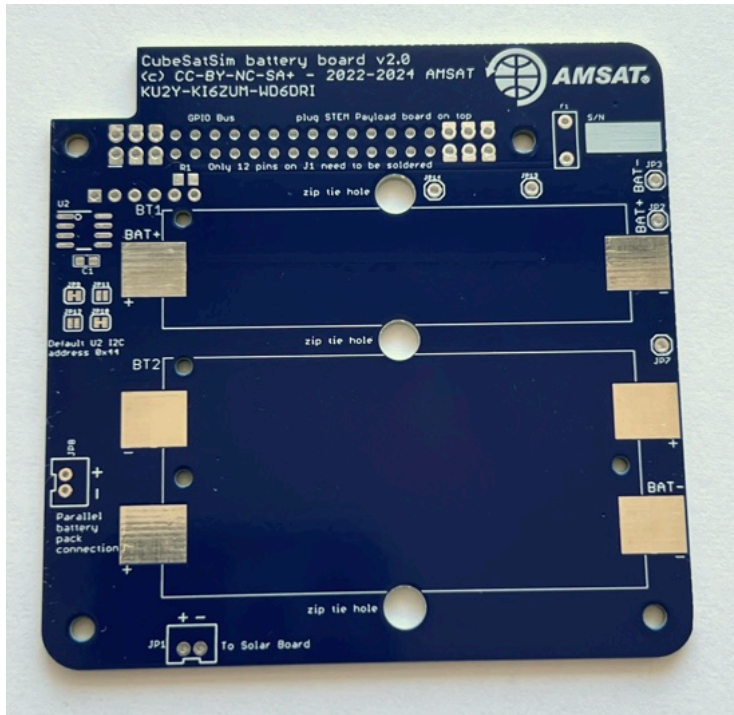
+ Add a custom sidebar

Clone this wiki locally

<https://github.com/alanbjohnston>

[https://github.com/alanjohnston/CubeSatSim/blob/42b777577a17e1c3bdc2f7bfbf3d5b9bec08124d/hardware/v2.0/cubesatsim-battery-v2.0.1\\_schematic.pdf](https://github.com/alanjohnston/CubeSatSim/blob/42b777577a17e1c3bdc2f7bfbf3d5b9bec08124d/hardware/v2.0/cubesatsim-battery-v2.0.1_schematic.pdf)

Here is the board with the top on the left with the AMSAT logo and the bottom on the right:



You will need these tools:

- Safety glasses (to protect eyes while soldering or trimming leads)
- Soldering iron and solder (I use lead-free solder, but leaded solder is easier to work with)
- Liquid flux, either in a bottle or pen

Other tools that are helpful:

- Multimeter (to read battery voltage)
- [Blue mounting putty](#) (to hold components in place while soldering)












## Checklist

---

The BOM has a sheet "By Steps" which lists the parts needed for each step in order.

[https://docs.google.com/spreadsheets/d/1Ta5UaJcinGozcheROrkfwXdGSDUZrXvQ1\\_nbIBdlIOY/e/dit?usp=sharing](https://docs.google.com/spreadsheets/d/1Ta5UaJcinGozcheROrkfwXdGSDUZrXvQ1_nbIBdlIOY/e/dit?usp=sharing) If you have a Google account, you can make a copy of this spreadsheet ("File" then "Make a Copy") and check off each part as you install it.

For example, here is the checklist for this step:

		Step 5. Battery Board	<a href="https://github.com/alanbjoh">https://github.com/alanbjoh</a>		
<input checked="" type="checkbox"/>	Ref	Item	Qty	Location	Image
<input type="checkbox"/>		Battery PCB	1		
<input type="checkbox"/>	BT1	1 cell AA battery holder 1024		Top	
<input type="checkbox"/>	BT2	2 cell AA battery holder 1012		Top	
<input type="checkbox"/>	U1	Blue INA219 High Side DC Current Sensor Br	1	Bottom	
<input type="checkbox"/>	J1	GPIO 20x2 female stacking header extra long	1	Bottom	
<input type="checkbox"/>	JP1	Micro JST 2 pin connector	1	Top	
<input type="checkbox"/>		Tenergy AA 2500mAh NiMH Rechargeable B	1	Top	
<input type="checkbox"/>		dual battery clip	1	Top	
<input type="checkbox"/>		single battery clip	1	Top	
<input type="checkbox"/>		zip tie	2		
<input type="checkbox"/>		JST jumper cable	1		

## Battery Board Instructions

### Video

Here is a [video of this step](#).

### Assembly

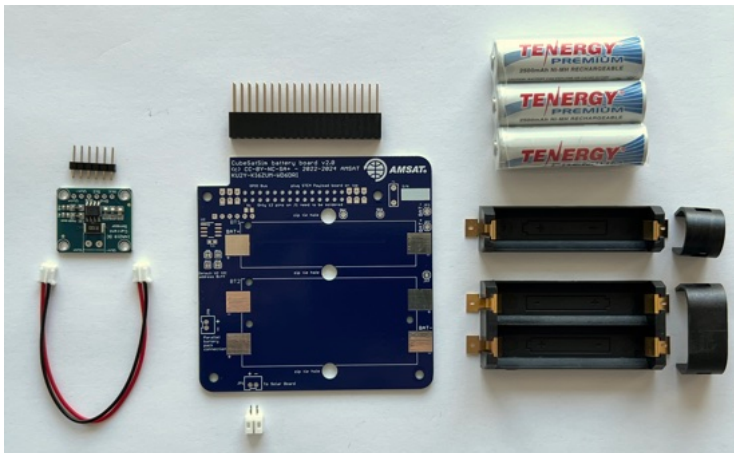
The Battery Board stacks on top of the Pi Zero and under the Main Board.

You will need the following parts to make the Battery board as described in the BOM

<https://cubesatsim.org/bom>:

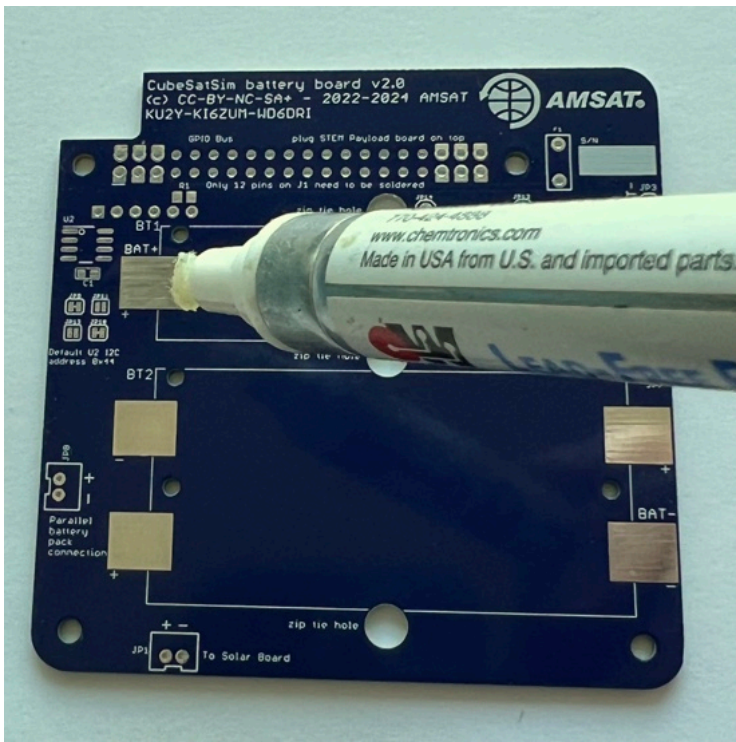
- Battery PCB
- Stacking GPIO header J1
- Three Nickel Metal Hydride (NiMH) cells (AA size)
- Battery holders (AA size) BT1 and BT2
- INA219 blue voltage and current sensor board U1
- JST 2.0 jumper cable
- JST 2.0 connector JP1
- Two small zip ties

The parts are shown here:

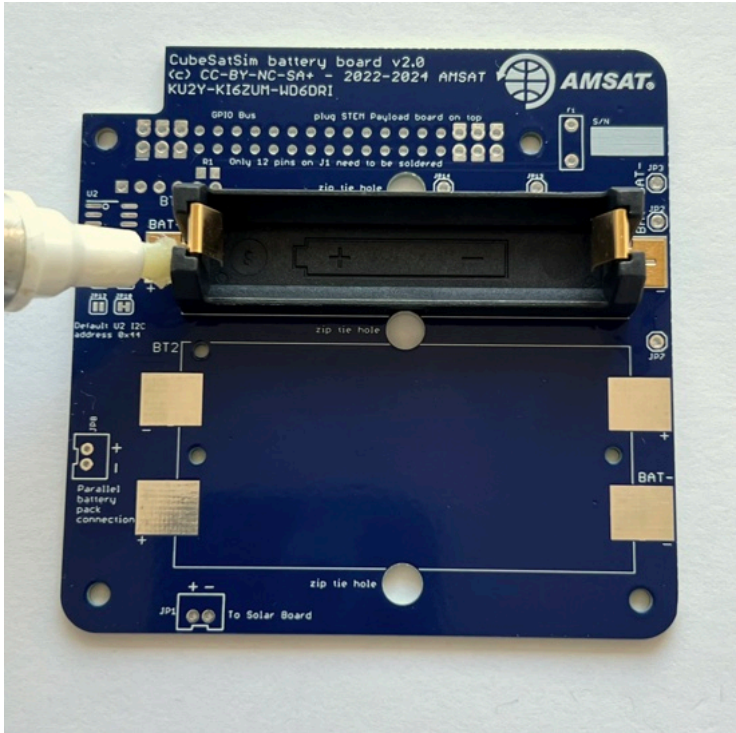


On the top of the PCB, apply liquid flux to the six pads for the battery holders:

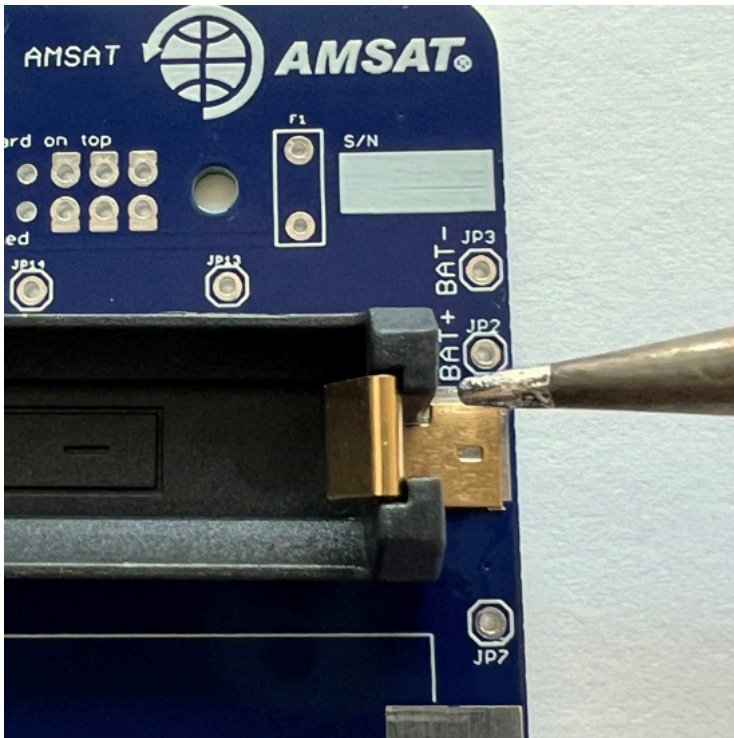




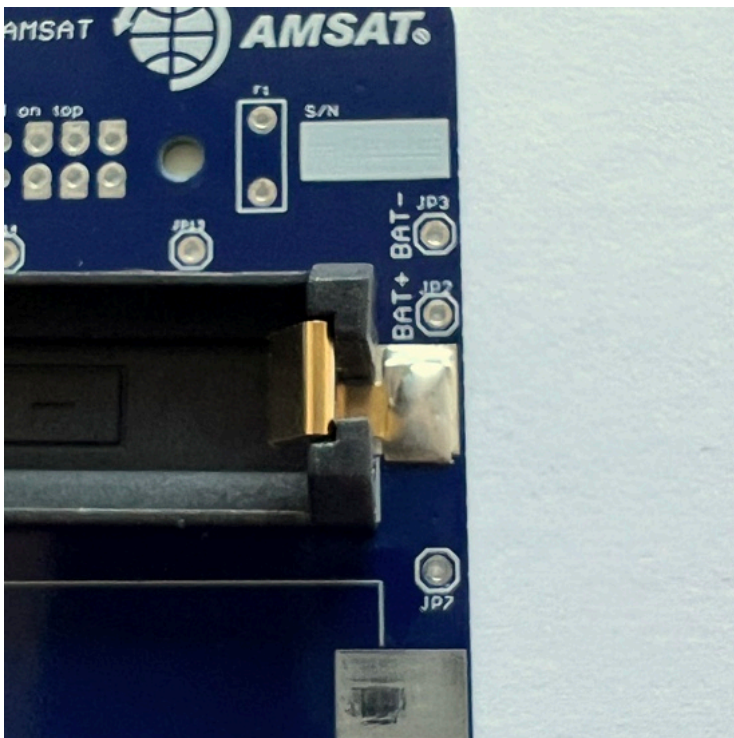
Then mount the battery holder BT1. There is a small tab on the bottom of the battery holders that fits in a hole in the PCB - make sure it is inserted so the battery polarity is correct and the holder is flat against the PCB. Apply more liquid flux on part:



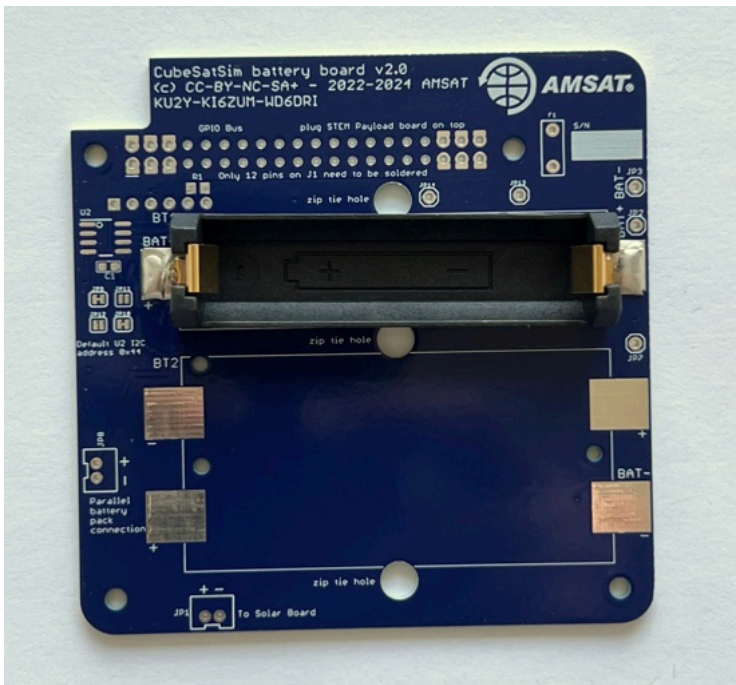
When soldering, make sure you apply some heat to the PCB, not just the part. This will ensure that the solder flows around and under the part



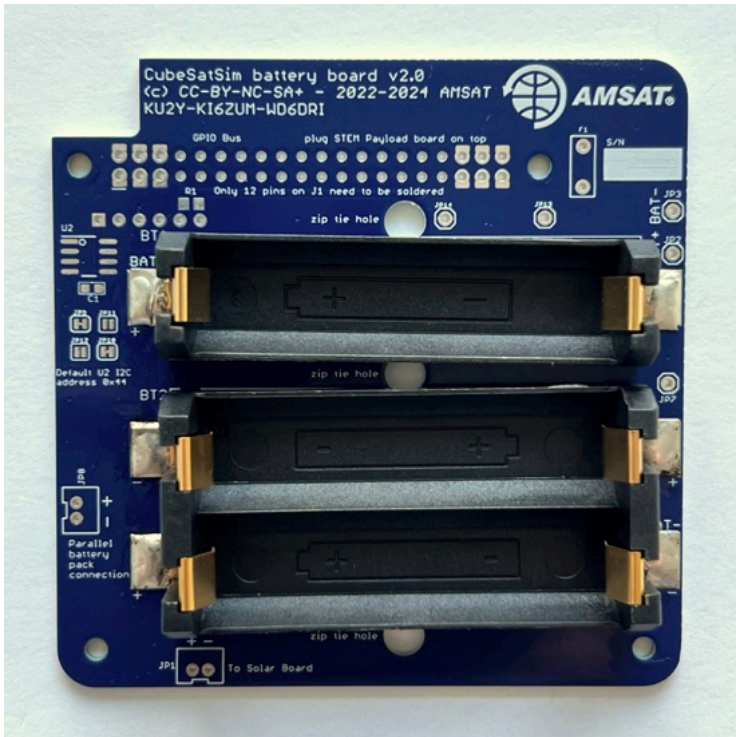
Here's how it looks when the pad is soldered:



Solder the other pad (note this part is a little crooked):



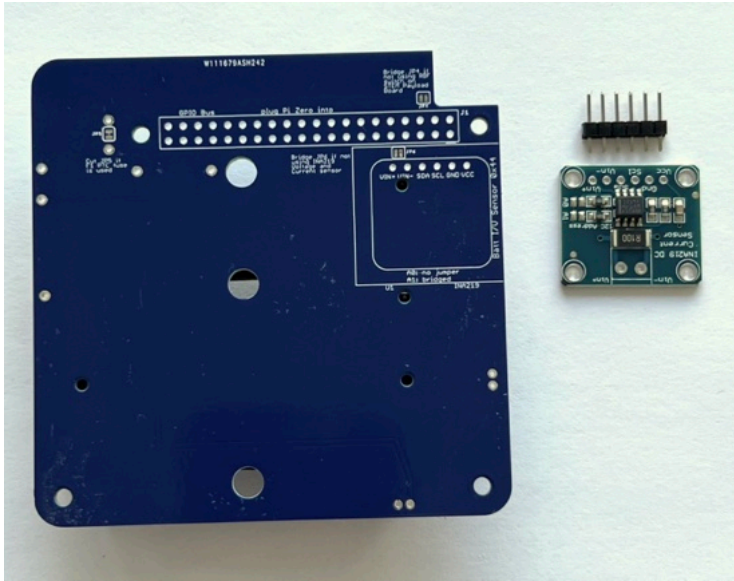
Then mount battery holder BT2:



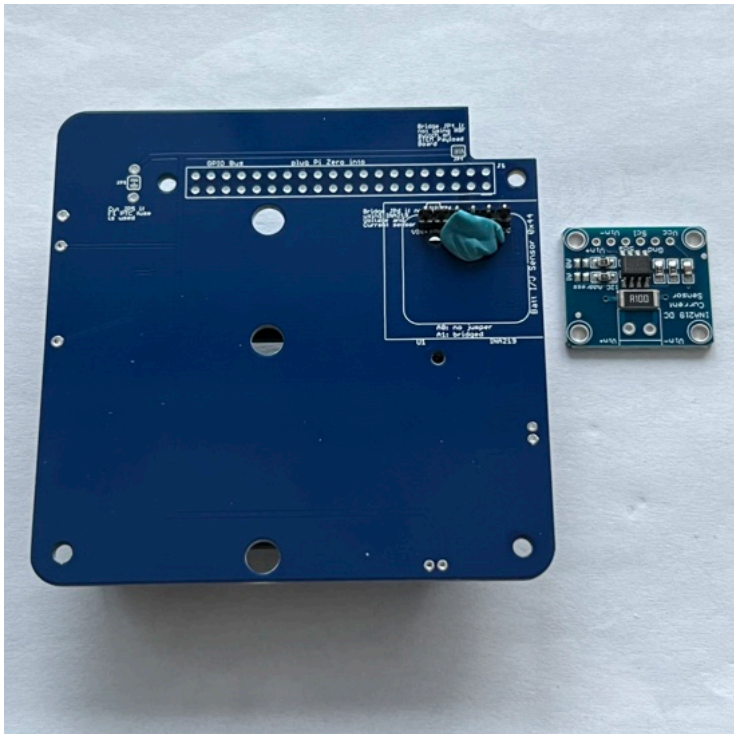
Again, there is a small tab on the bottom of the battery holders that fits in a hole in the PCB - make sure it is inserted so the battery polarity is correct and the holder is flat against the PCB.



Next, turn the PCB upside down, as the blue INA219 board is mounted on the bottom of the PCB:



Insert the 1x6 male pin header into the PCB and hold in place with blue putty:



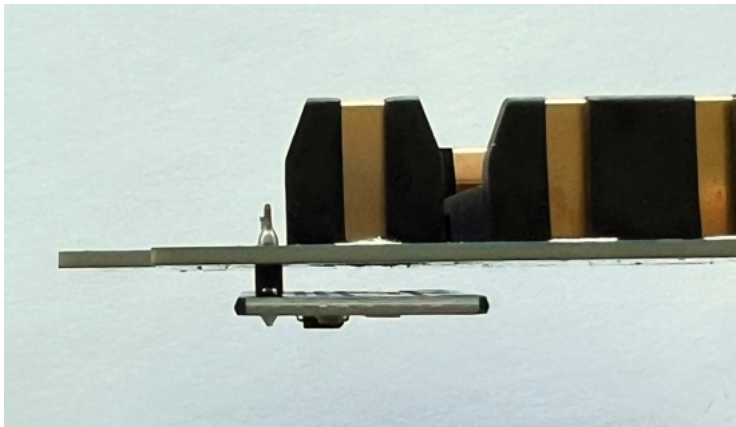
Solder the six pins on the other side.



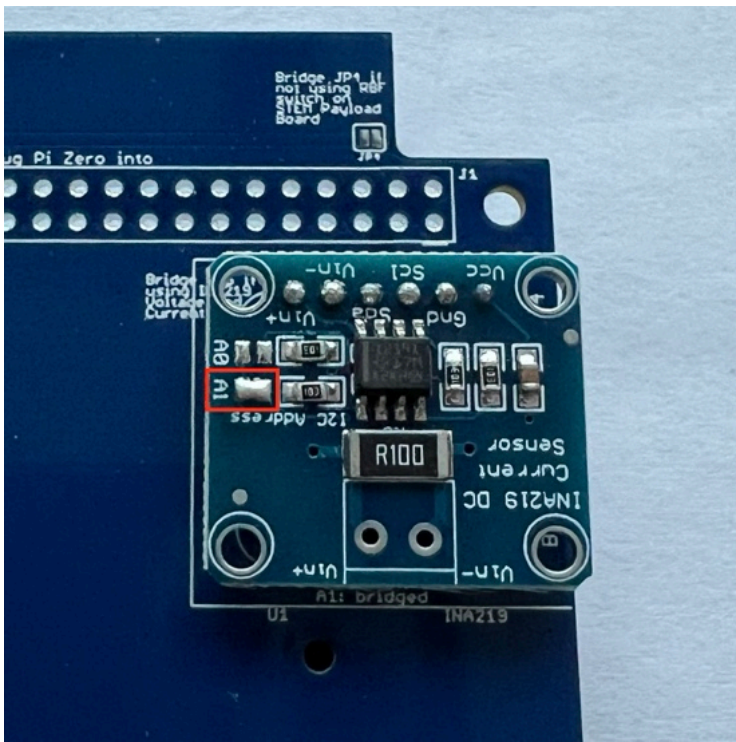




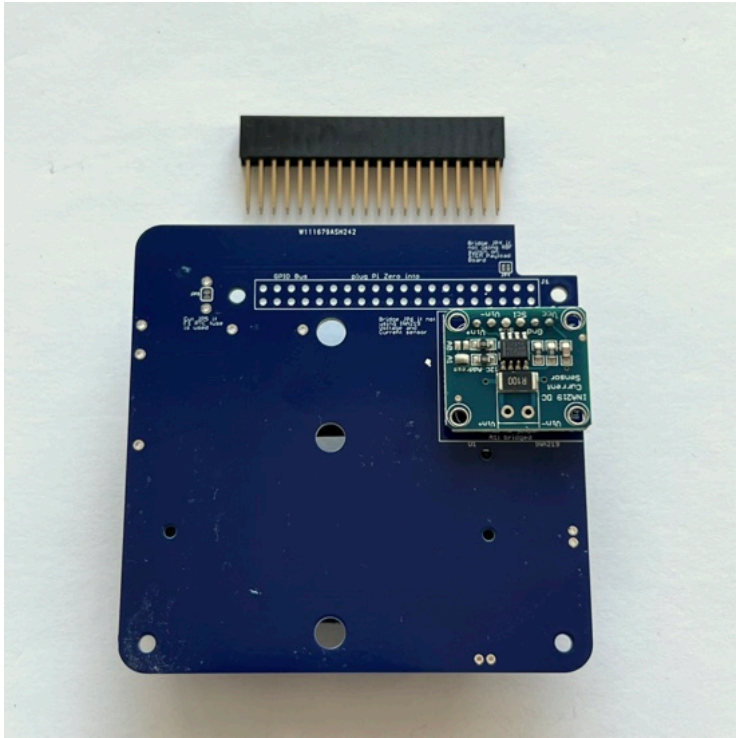
Here's how it looks from the side:



The jumper A1 needs to be bridged with a blob of solder to set the I2C address for the board:

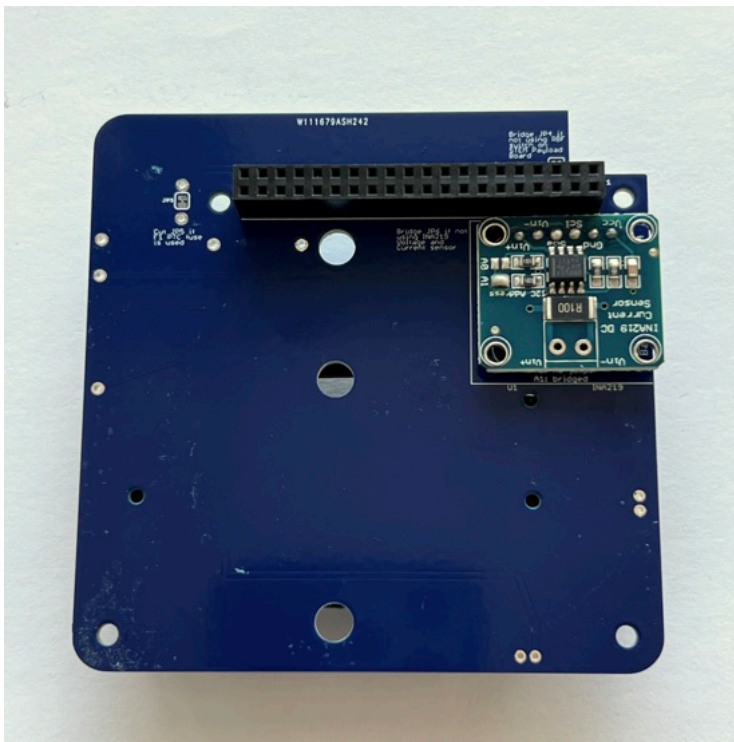


Next, also on the bottom of the PCB, mount the stacking GPIO header J1:

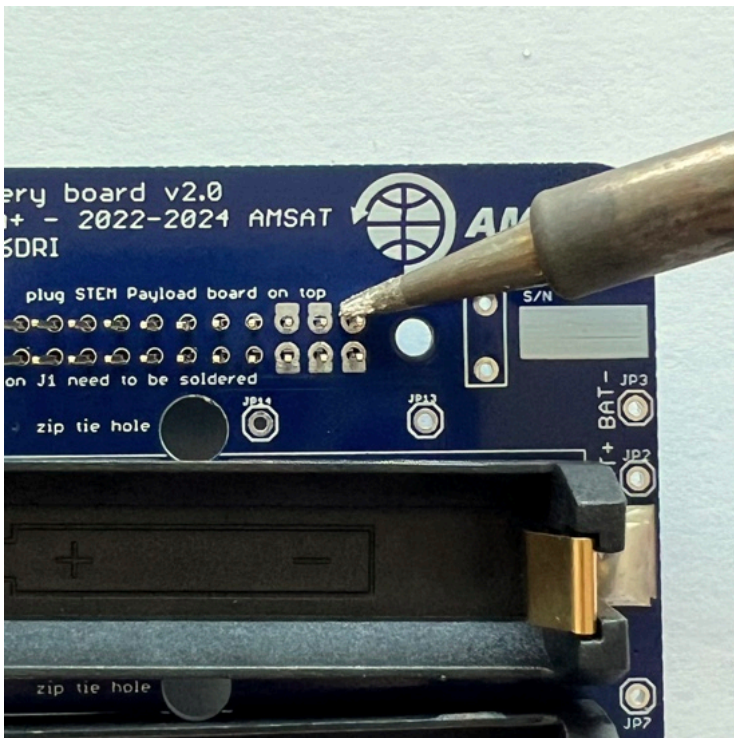


Insert it on the bottom as shown:



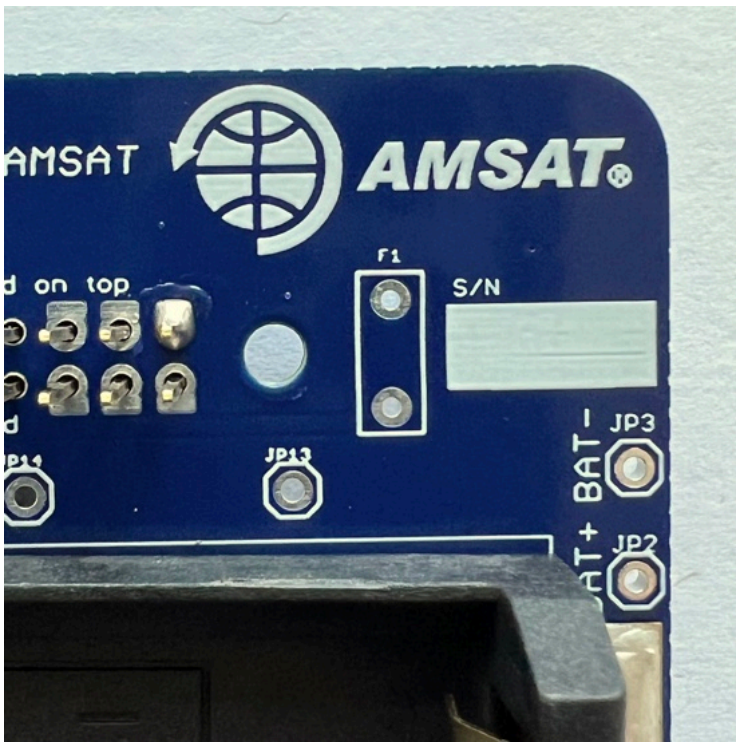


Flip the PCB to the top. Solder one pin on either side, using the pad next to each pin to heat both the pin and the pad:

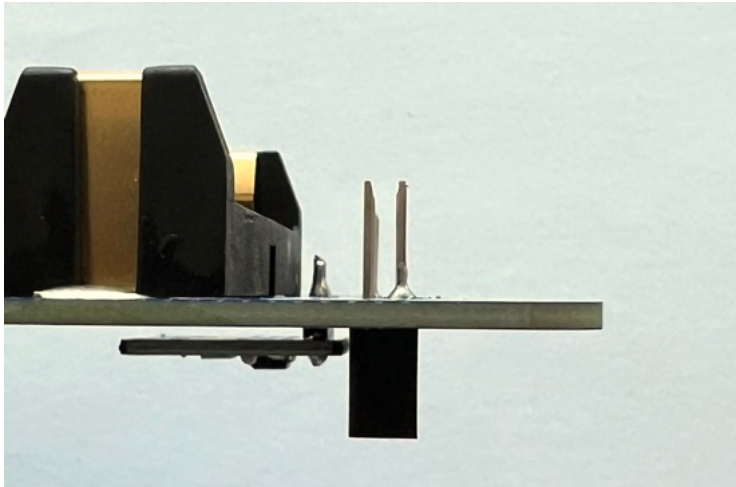


Make sure you don't get solder on the upper part of the pin or it won't insert into the other board:

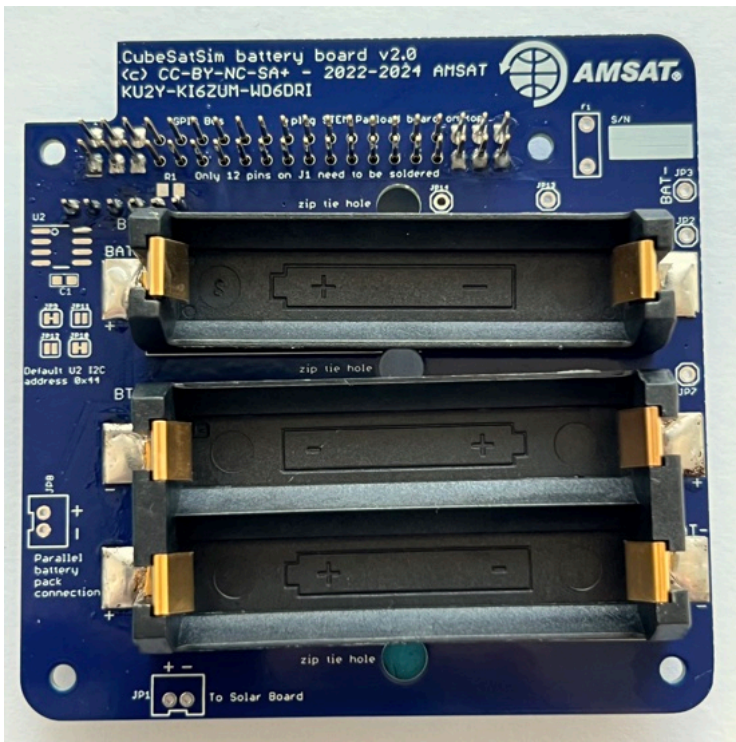




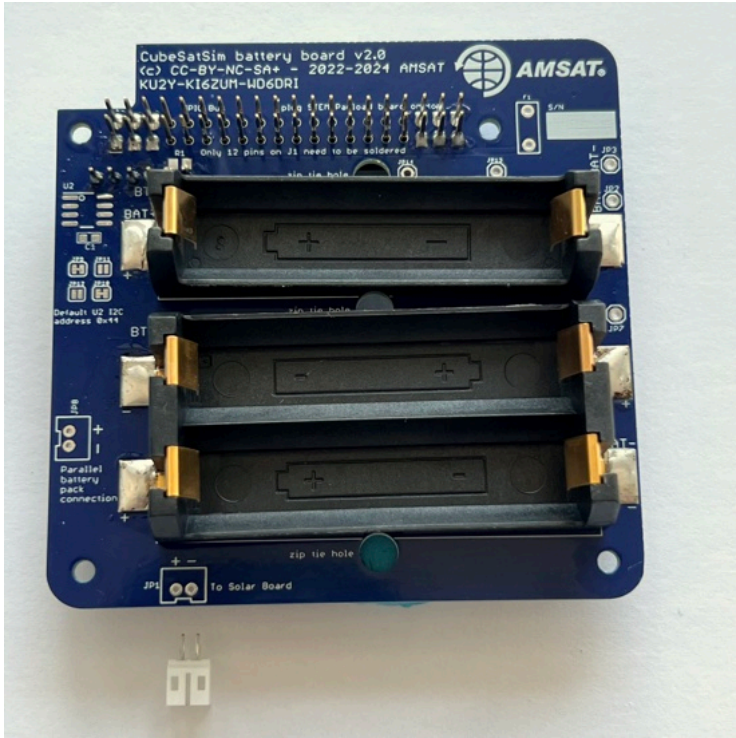
Make sure the GPIO header is fully inserted and straight and on the correct side:



Only solder six pins on either side, as shown in the next photo:



Next, on the top of the PCB, insert the JST connector JP1.



Make sure the slot is facing the edge of the PCB and hold in place with blue putty:



Solder the pins on the other side:

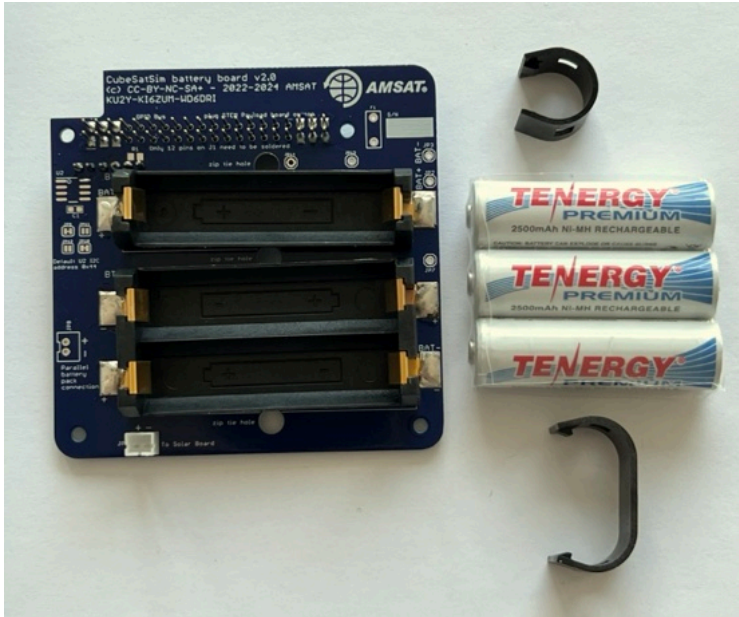


Here's how it looks from the side:





The three NiMH batteries can be inserted and held in place with the clips or zip ties.



First plug in the batteries, paying attention to the polarity:



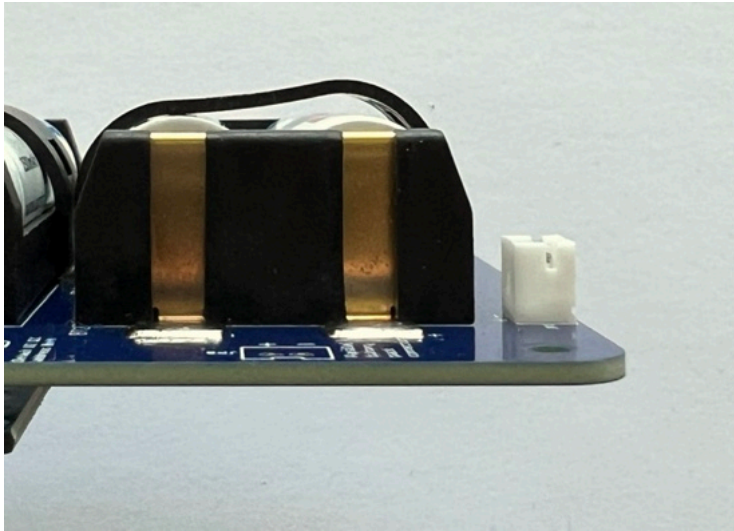


It is a good idea to test your battery polarity using a voltmeter. Use the BAT+ (J1) and BAT- (J2) test points on the board, being very careful not to short them together:

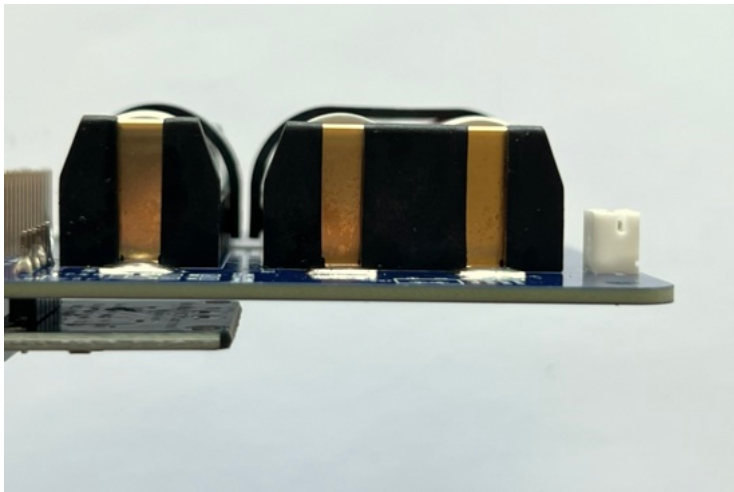


If you read a positive voltage in the range 3V - 4.5V, your Battery board is "nominal" and ready to be used. If you get a negative voltage, check that the red and black test leads are plugged into the positive and common inputs on your meter, or that your batteries aren't inserted backwards.

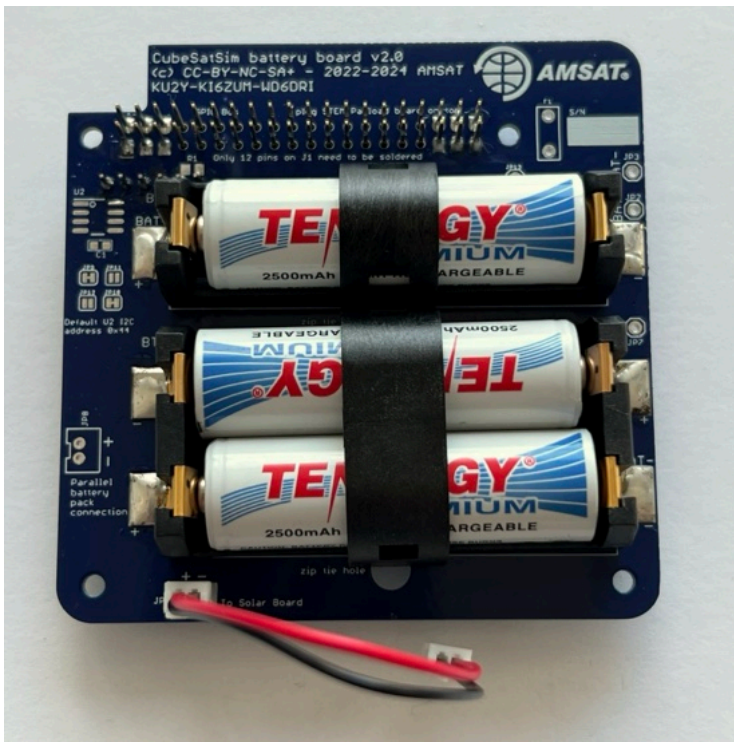
Secure the batteries with the clips. The clips need to be pushed down quite hard to get them to lock. This clip is not locked:



These clips are locked:



Here's how the completed board looks with the JST jumper wire connected to the JST connector:



The next step is to [assemble the Solar Board](#).

+ Add a custom footer

# 6. Solar Board

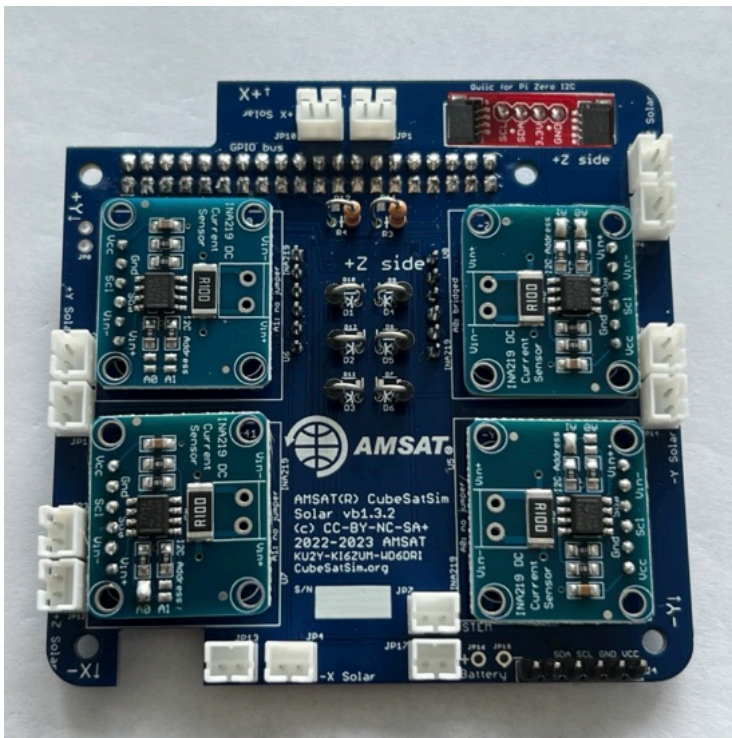
[Edit](#) [New page](#)

Alan Johnston edited this page 5 days ago · [2 revisions](#)

If the images on this page fail to load, you can [download a PDF of the page here](#).

## 6. Solar Board

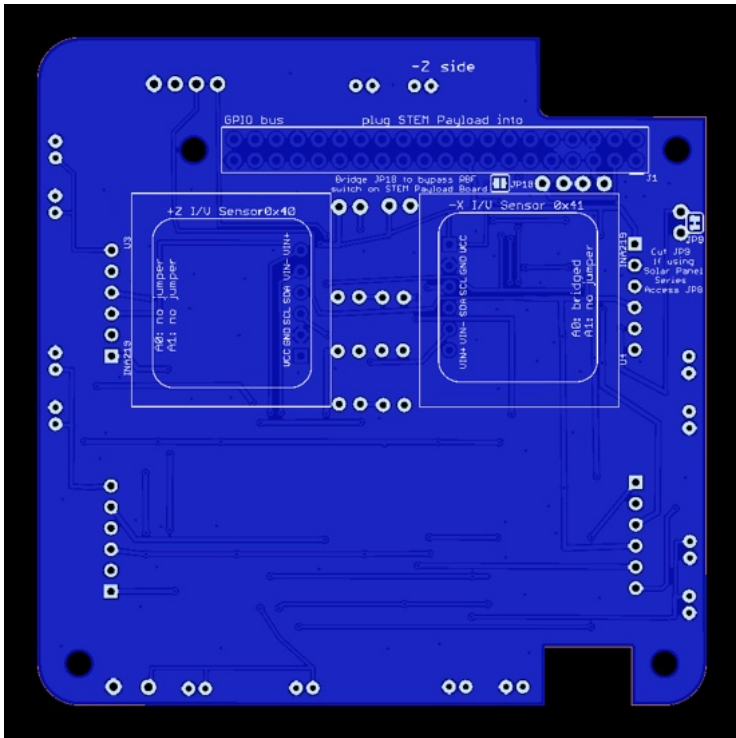
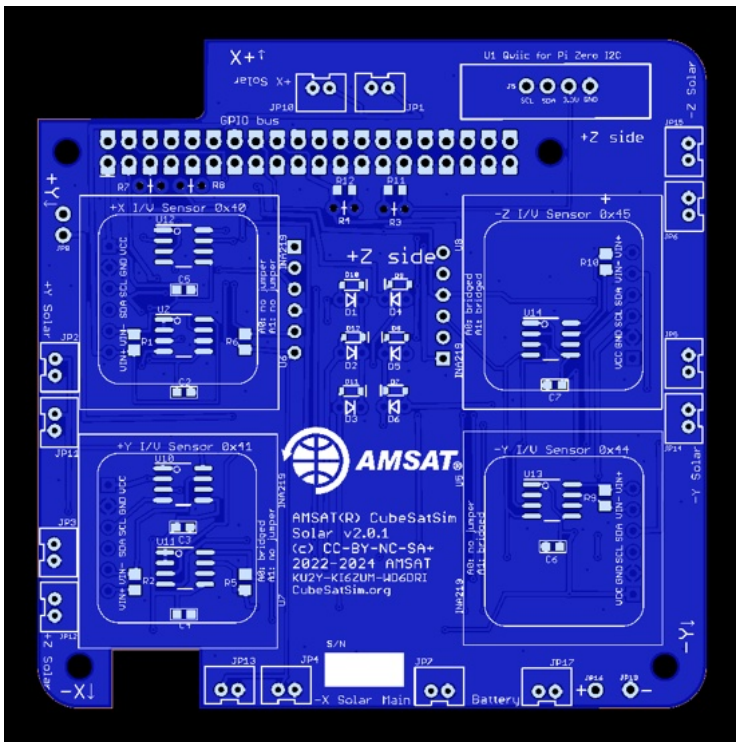
These steps are to build the CubeSatSim Solar PCB:



Here is the Solar PCB v2.0:

- Pages 147
- Find a page...
- Home
- 1. Main Board 1
- 2. Software Install
- 3. Ground Station
- 4. Main Board 2
- 5. Battery Board
- 6. Solar Board
  - 6. Solar Board
  - Checklist
  - Video
  - Steps
- 7. Solar Panels and Frame
- 8. Board Stack
- 9. Final Testing
- Adding New Sensors
- Command and Control





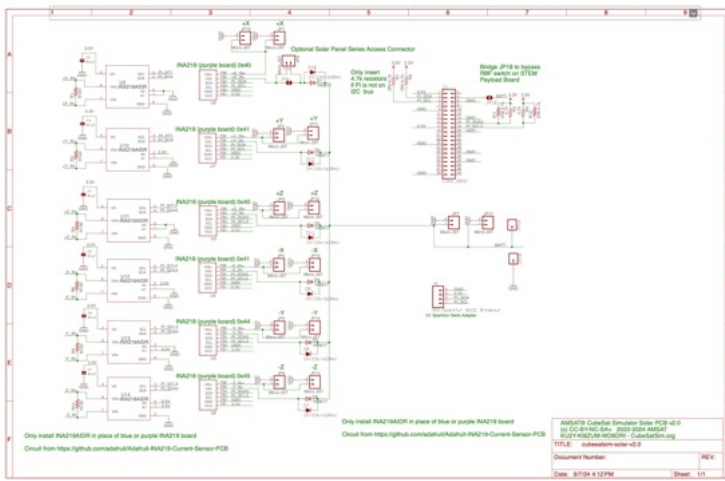
Here is the schematic:

- ▶ [Creating the CubeSatSim Raspber...](#)
  - ▶ [CubeSatSim Lite](#)
  - ▶ [CubeSatSim Loaner User Guide](#)
- Show 132 more pages...

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/alanbjohnston>



[https://github.com/alanbjohnston/CubeSatSim/blob/42b777577a17e1c3bdc2f7bfbf3d5b9bec08124d/hardware/v2.0/cubesatsim-solar-v2.0\\_schematic.pdf](https://github.com/alanbjohnston/CubeSatSim/blob/42b777577a17e1c3bdc2f7bfbf3d5b9bec08124d/hardware/v2.0/cubesatsim-solar-v2.0_schematic.pdf)

Note that these instructions show the beta v1.3.2 board, but it is identical to the v2.0 board.

You will need these tools:

- Safety glasses (to protect eyes while soldering or trimming leads)
- Soldering iron and solder (I use lead-free solder, but leaded solder is easier to work with)
- Needle nose pliers (to bend leads and hold parts)
- Side cutters (to trim leads)
- glue, such as super glue gel to attach the JST connectors to the PCB

Other tools that are helpful:

- [Blue mounting putty](#) (to hold components in place while soldering)











## Checklist

---

The BOM has a sheet "By Steps" which lists the parts needed for each step in order.

[https://docs.google.com/spreadsheets/d/1Ta5UaJcinGozcheROrkfwXdGSDUZrXvQ1\\_nbIBdlIOY/e/dit?usp=sharing](https://docs.google.com/spreadsheets/d/1Ta5UaJcinGozcheROrkfwXdGSDUZrXvQ1_nbIBdlIOY/e/dit?usp=sharing) If you have a Google account, you can make a copy of this spreadsheet ("File" then "Make a Copy") and check off each part as you install it.

For example, here is the checklist for this step:

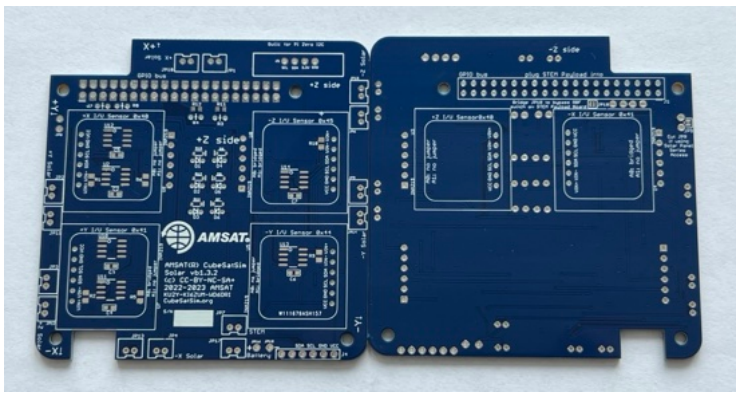
Step 6. Solar Board		<a href="https://github.com/alanbj">https://github.com/alanbj</a>			
<input checked="" type="checkbox"/>	Ref	Item	Qty	Location	Image
<input type="checkbox"/>		Solar Board PCB	1		
<input type="checkbox"/>	D1-D6	1N5817 diode	6	Top	
<input type="checkbox"/>	R3, R4	4.7k resistors	2	Top	
<input type="checkbox"/>	J1	20x2 female GPIO header non-stacking	1	Bottom	
<input type="checkbox"/>	U3, U4	Blue INA219 High Side DC Current Sensor Br	2	Bottom	
<input type="checkbox"/>	U5, U6	Blue INA219 High Side DC Current Sensor Br	4	Top	
<input type="checkbox"/>	JP1-JP2	Micro JST 2 pin connectors	14	Top	
<input type="checkbox"/>		QWICC connector	1	Top	
<input type="checkbox"/>		1x4 male breakaway header	1	Top	
<input type="checkbox"/>		JST jumper cable	1	Top	

## Video

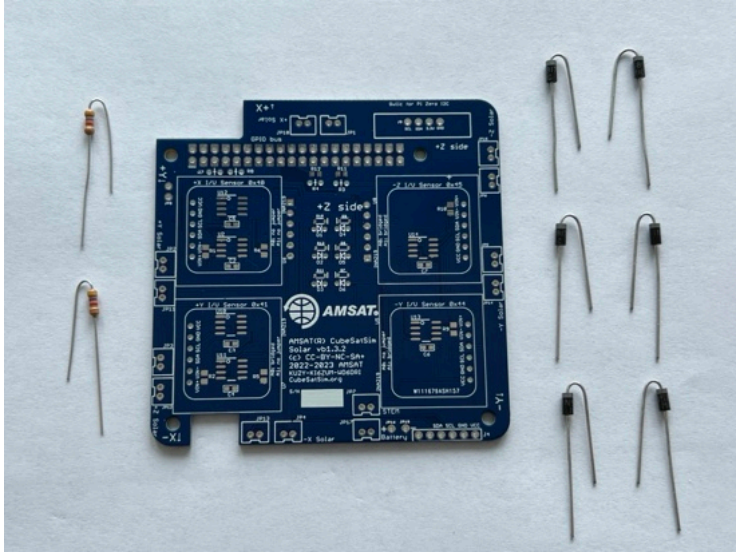
Here is a [video of this step](#).

## Steps

Start with the blank PCB, shown here top on left with AMSAT logo, bottom on the left:

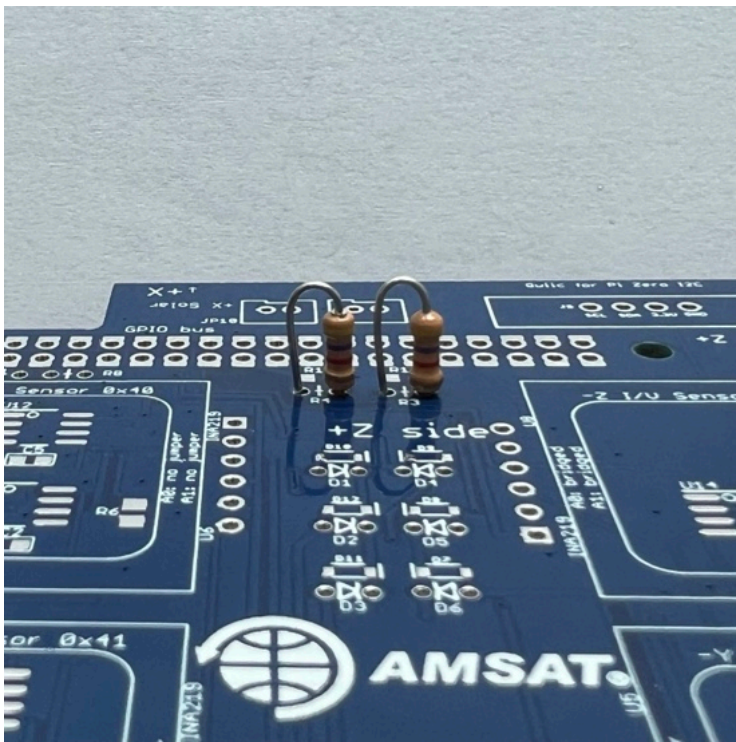


The 1N5817 diodes D1 - D6 and 4.7k Ohm resistors R3 and R4 are installed vertically:

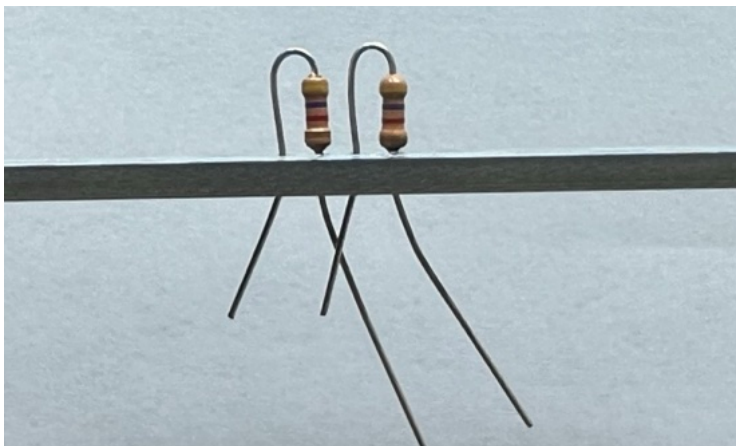


First insert the 4.7k Ohm resistors R3 and R4 (color bands yellow blue red):

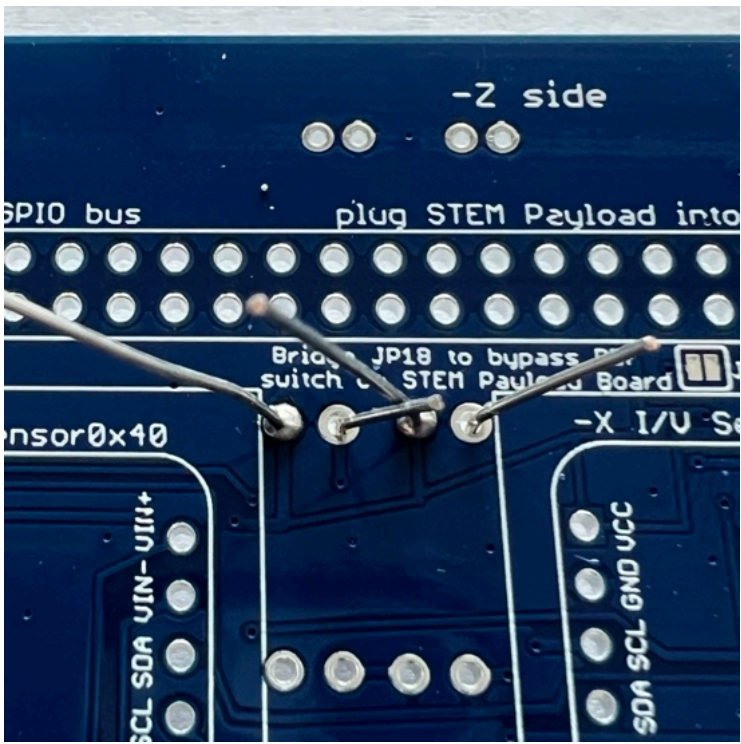




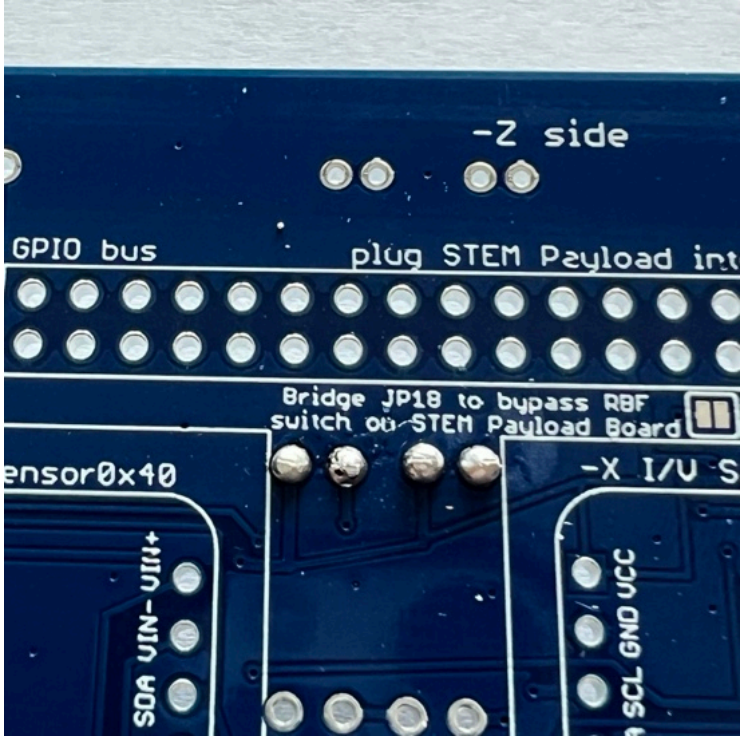
Bend the leads on the bottom of the PCB so they are held in place:

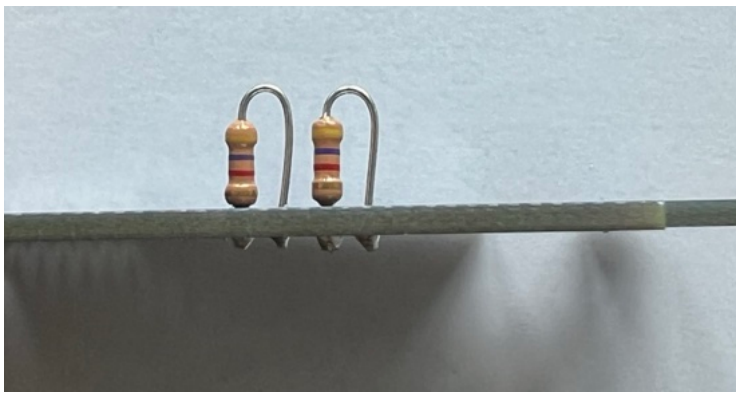


Solder one pin on each resistor then check the top of the PCB that the resistors are straight and inserted all the way:

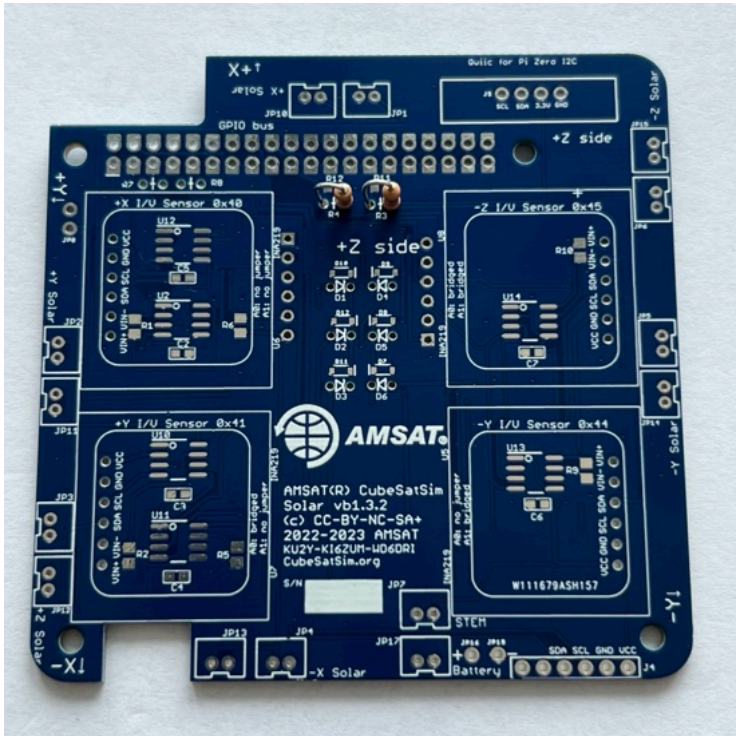


Then solder the other pin, the trim the leads:



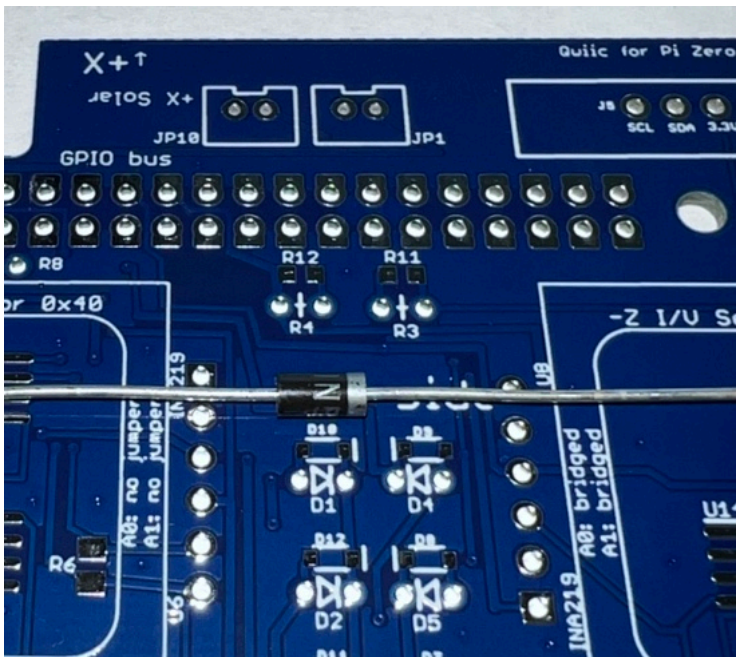


Here's how the board looks:

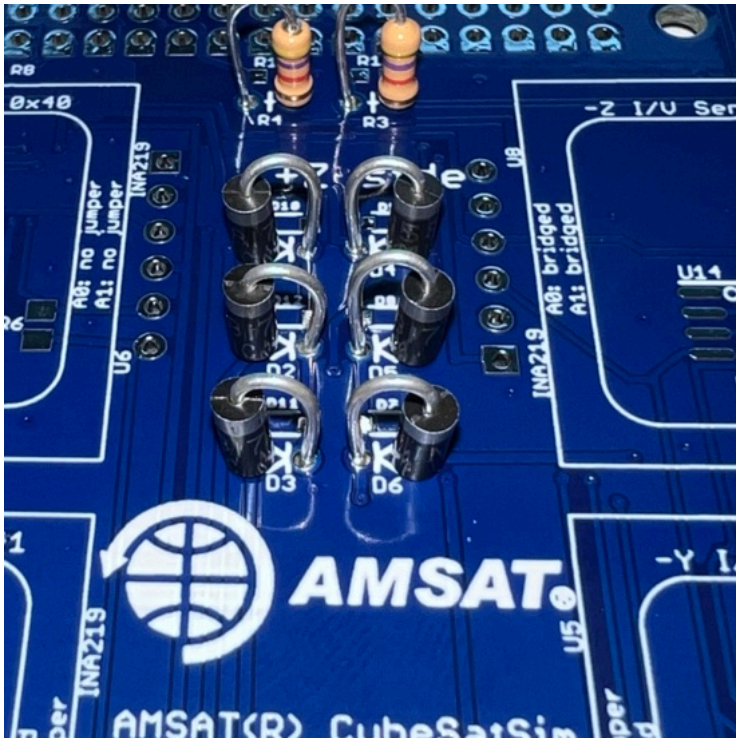


Next, solder the six diodes. The polarity of this diode is marked by a white band, which is on the right side in this photo, which matches the schematic symbol on the PCB for D1:



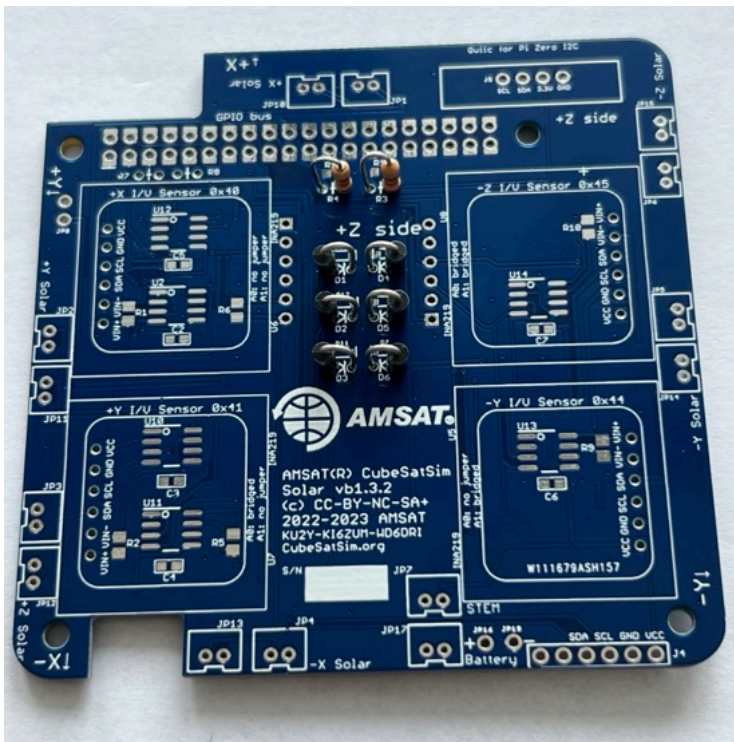


Here is the polarity of the diodes:

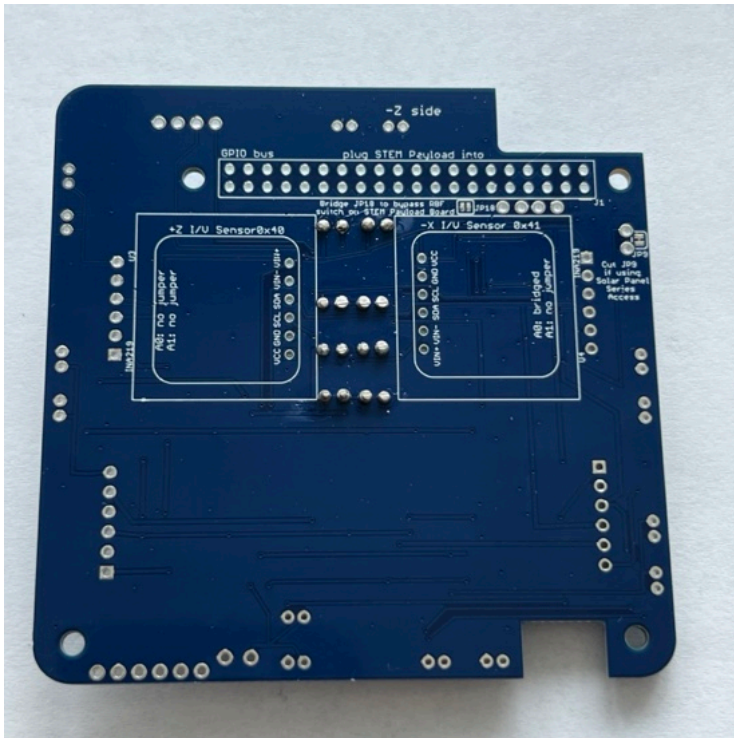


They are installed on the PCB top then soldered on the PCB bottom. Solder one lead on each diode then flip the board over and double check polarity, insertion, and straightness before flipping board back over and soldering the other lead:

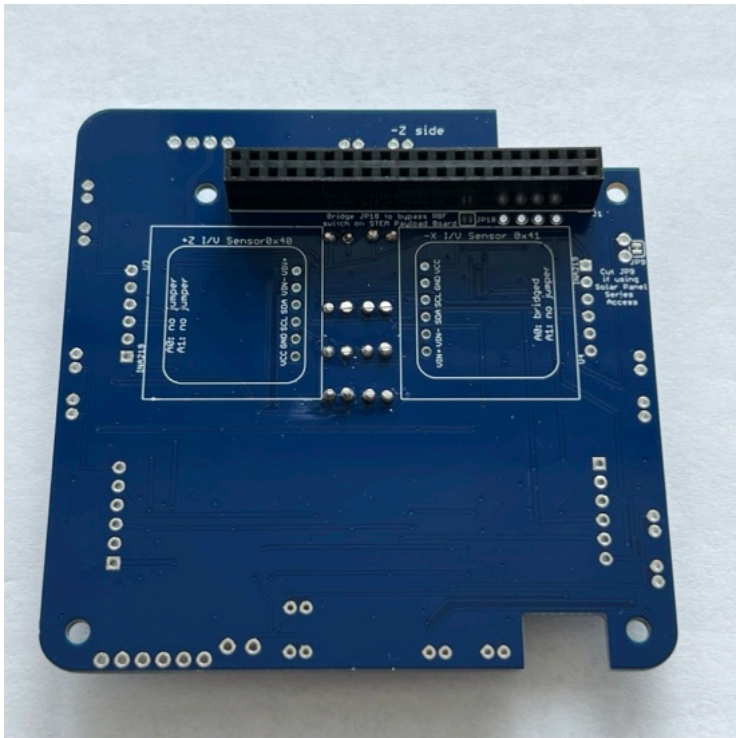
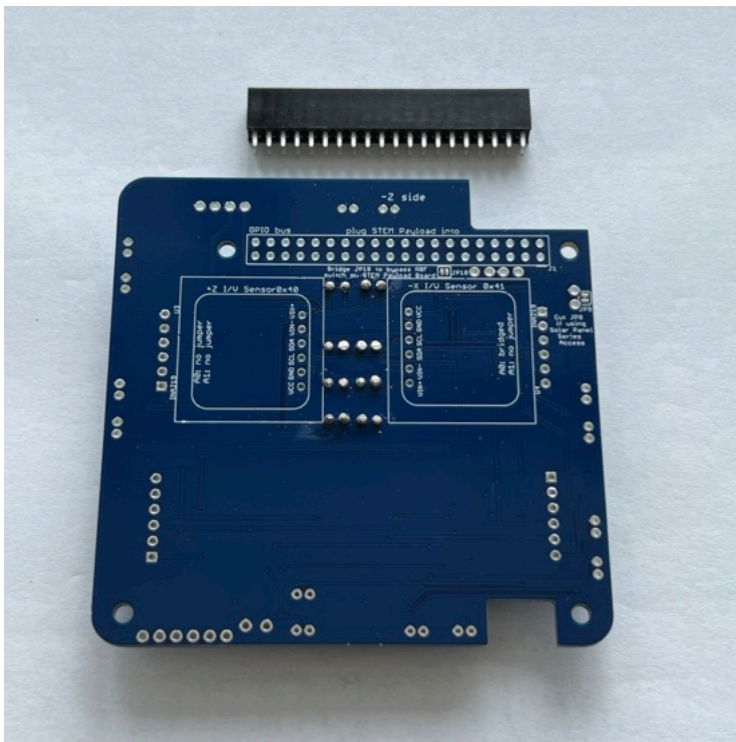




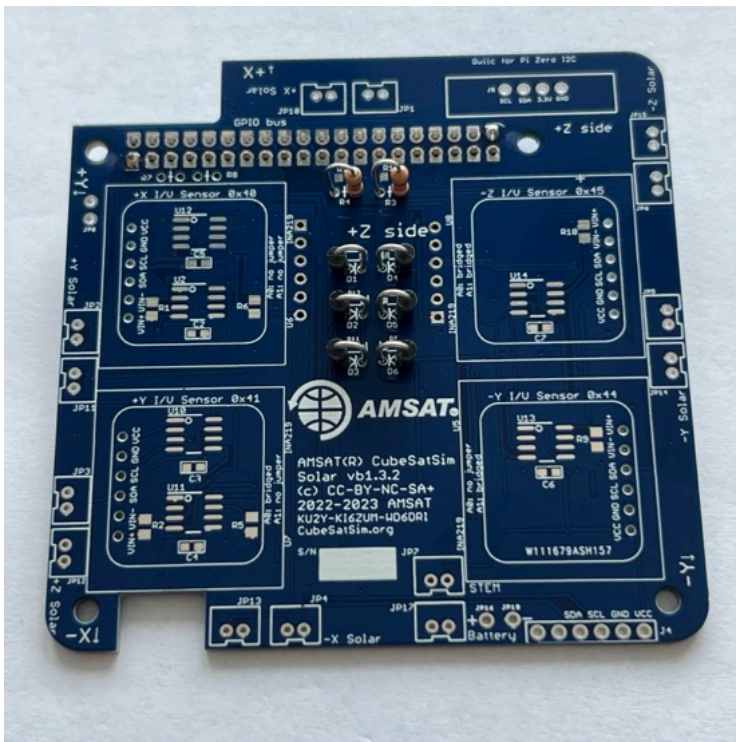
Trim the leads after soldering:



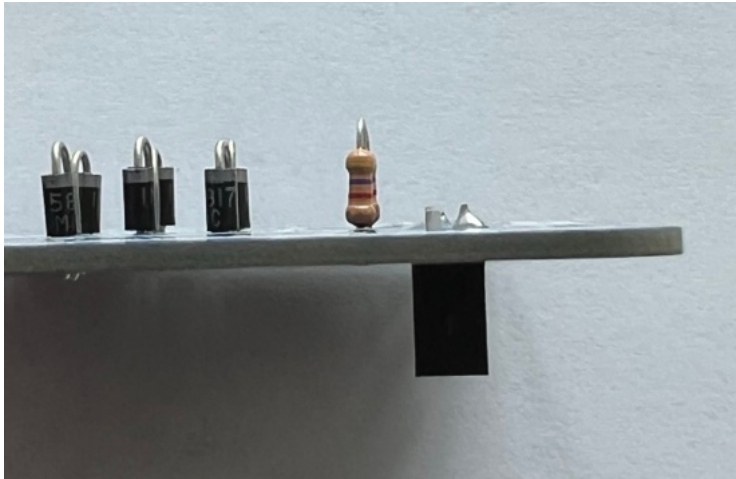
On the bottom, install the 2x20 GPIO female header J1.



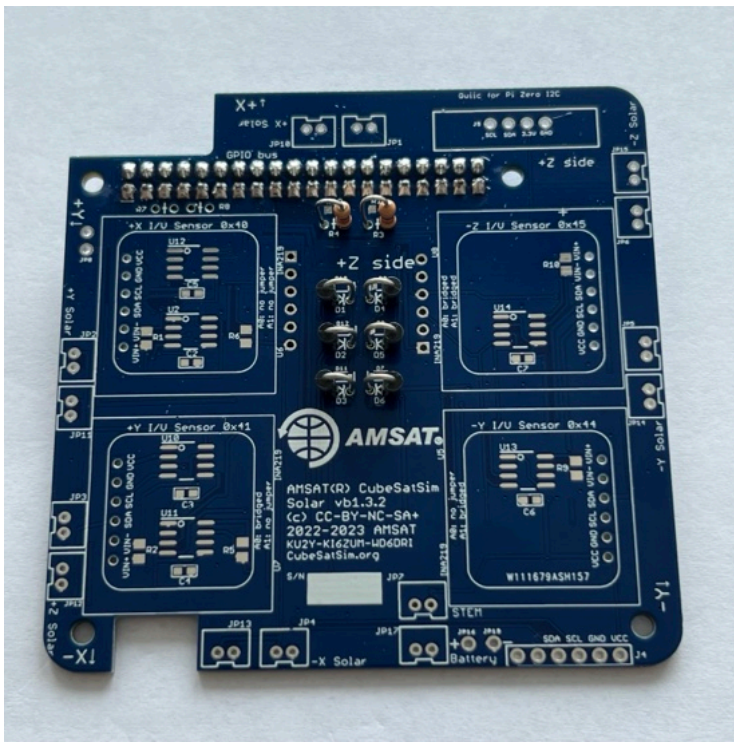
First solder only one pin on each side:



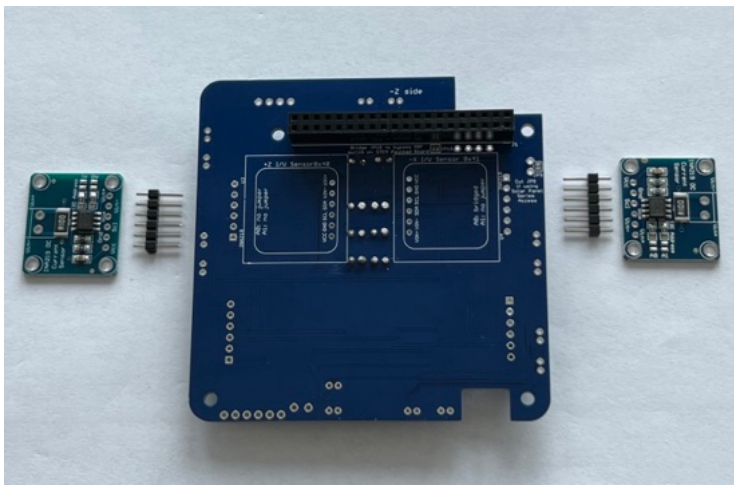
Verify it is fully inserted and straight before proceeding to solder the other pins:





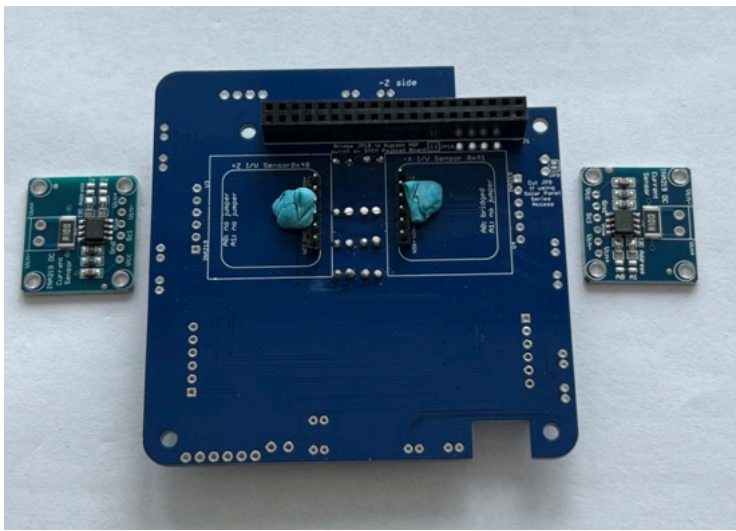


Also on the bottom, install two INA219 boards U3 and U4 and their 1x6 male pin headers:

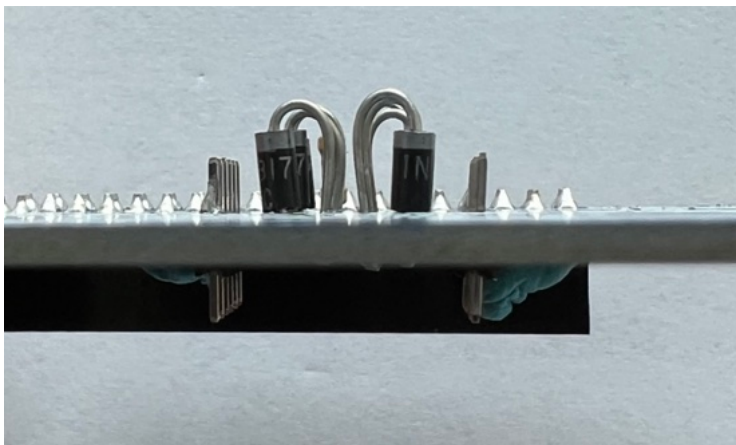
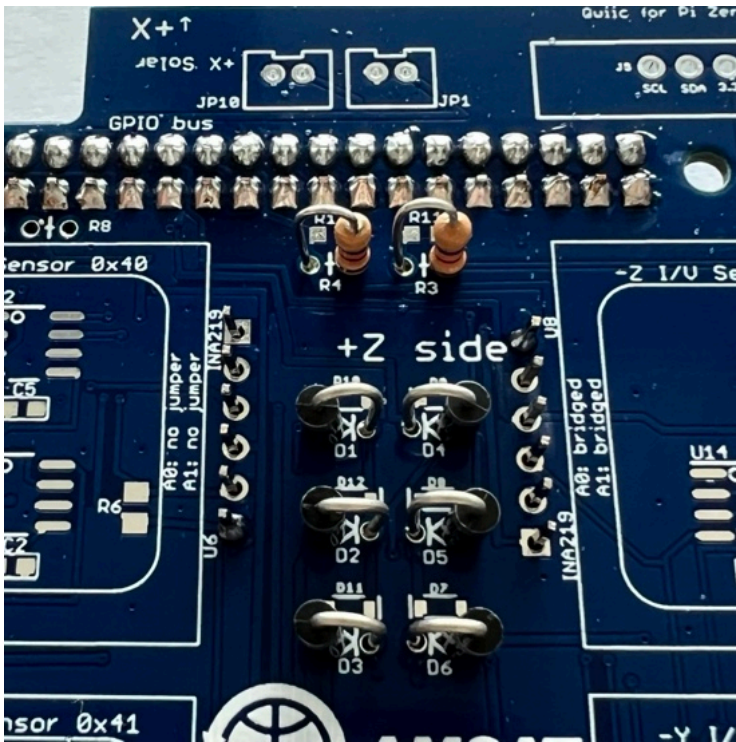


Insert the pin headers, long pins down and hold in place with blue putty while you solder them on the other side:

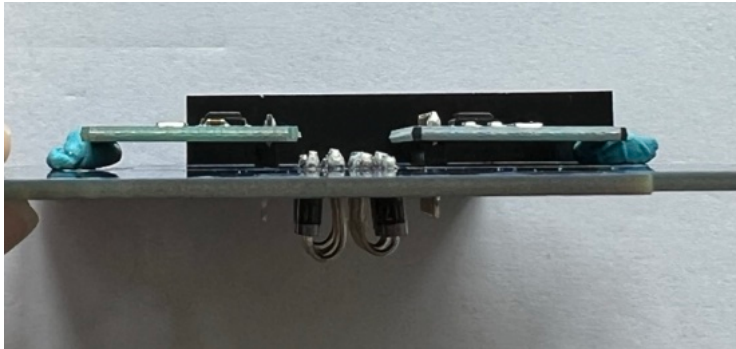
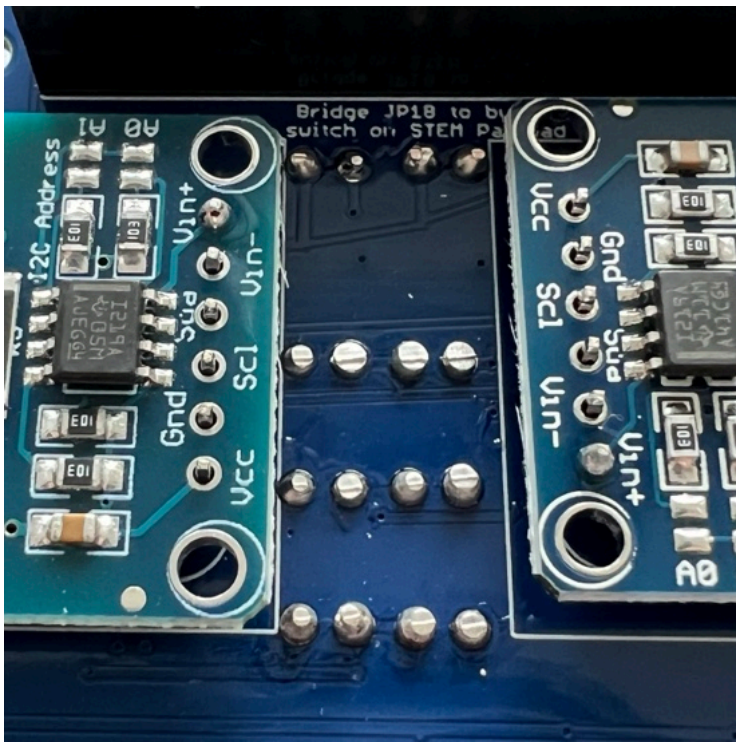




Solder one pin on each header, then check to make sure they are fully inserted and straight:

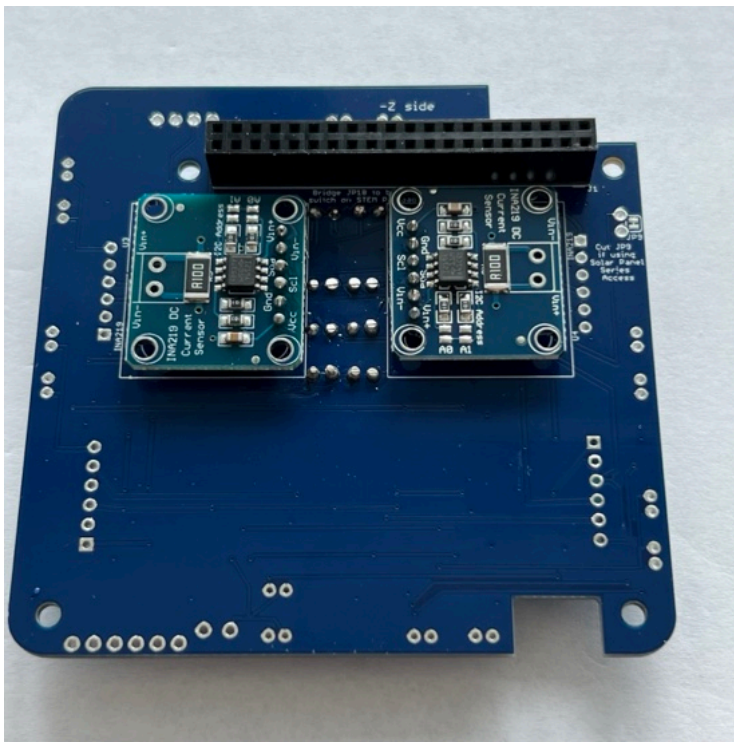




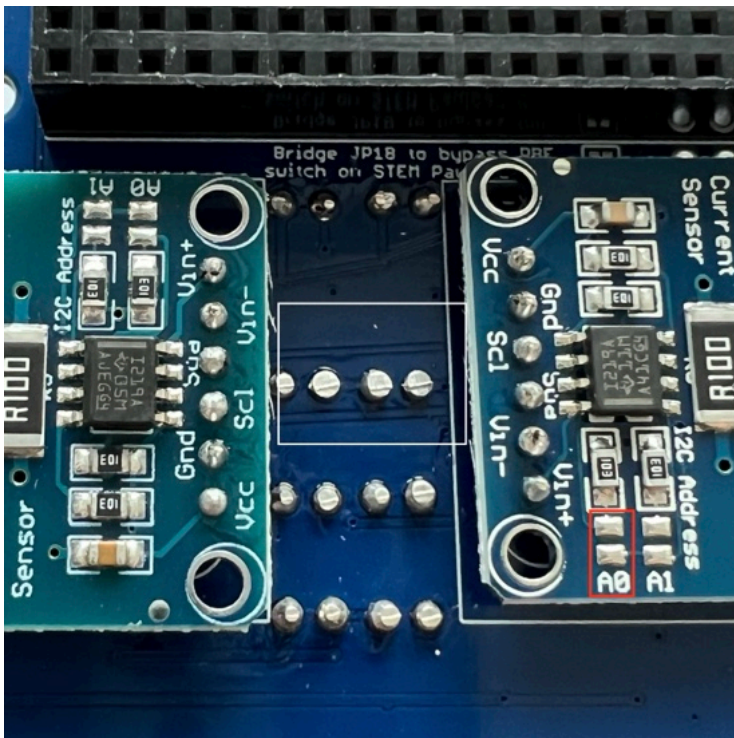


Solder the remaining pins:



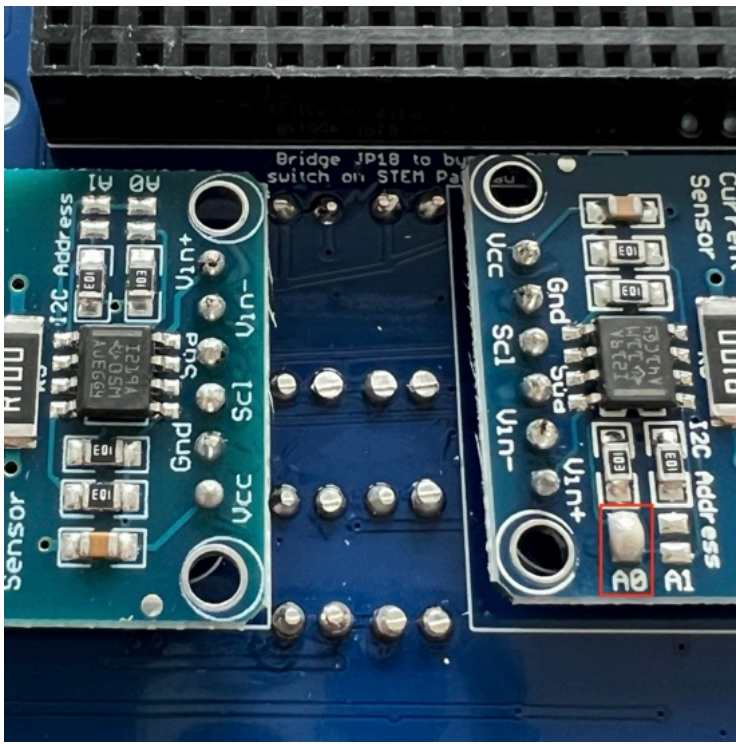


One jumper needs to be soldered on right INA219 (marked here in red):



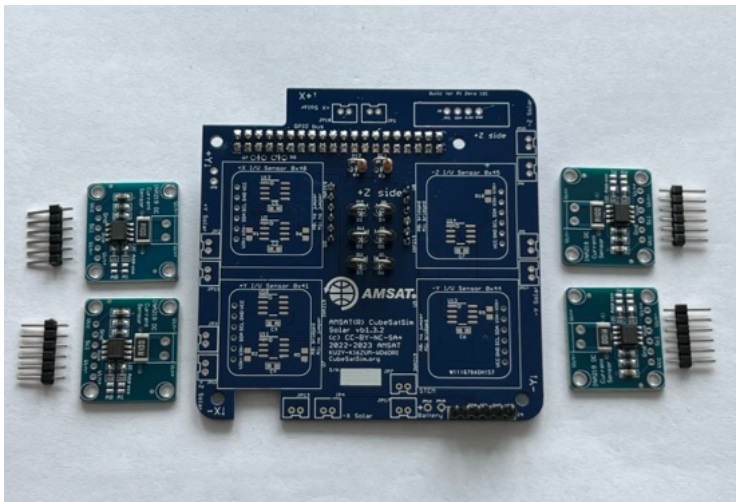
Add a small blob of solder on the pad so that it is bridged (shorted):



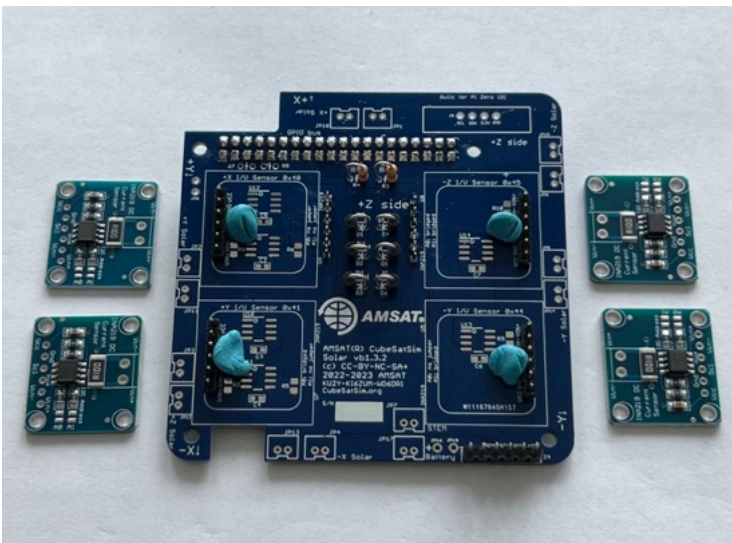


This completes the bottom of the PCB.

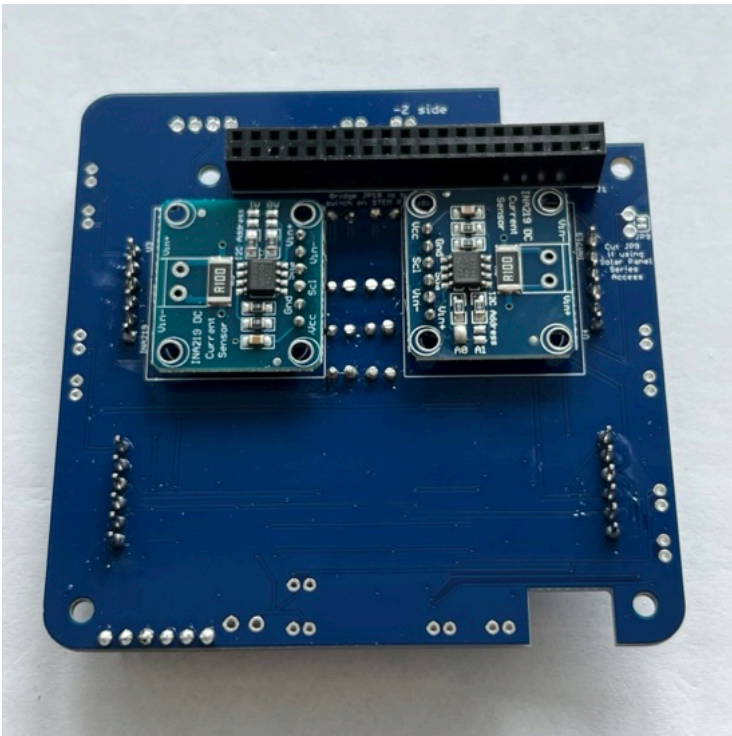
Flip the PCB to the top and install the four INA219 boards U5, U6, U7, and U8 and their 1x6 male pin headers.



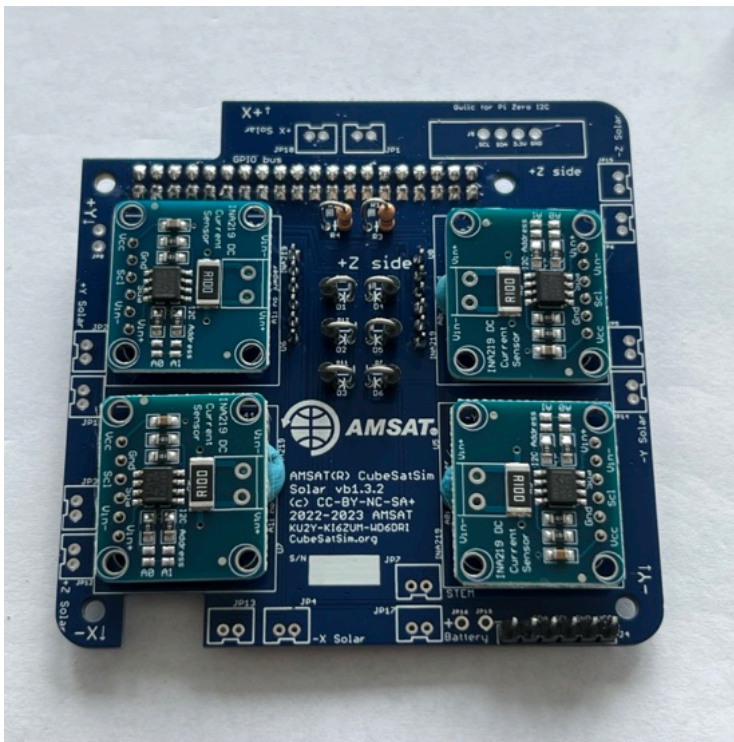
Insert the pin headers, the long side of the pins into the board. Hold in place with blue putty:



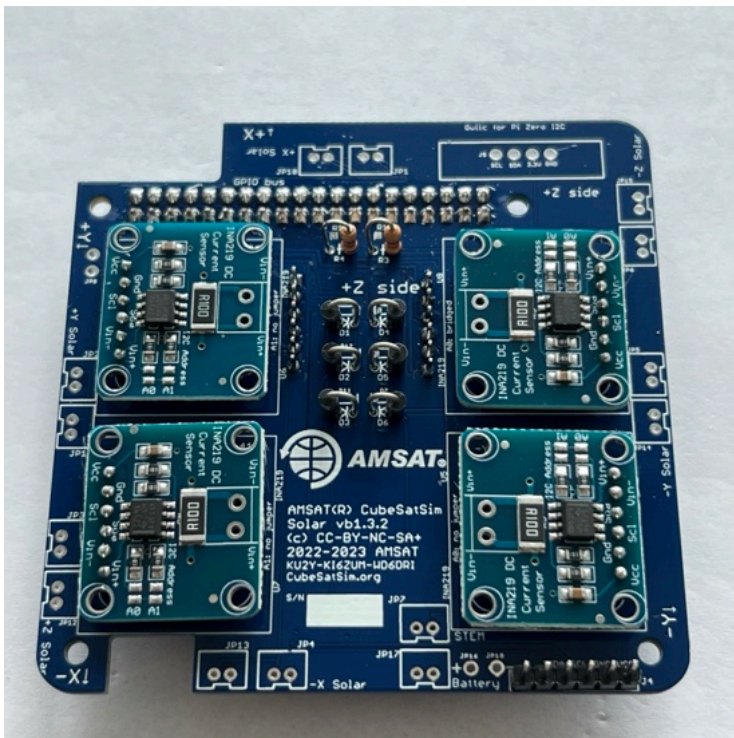
On the bottom of the PCB, solder one pin on each, then check for straightness. Then solder the rest of the pins:



On the PCB top, place the INA219 boards on the pin header and hold straight and in place with blue putty:

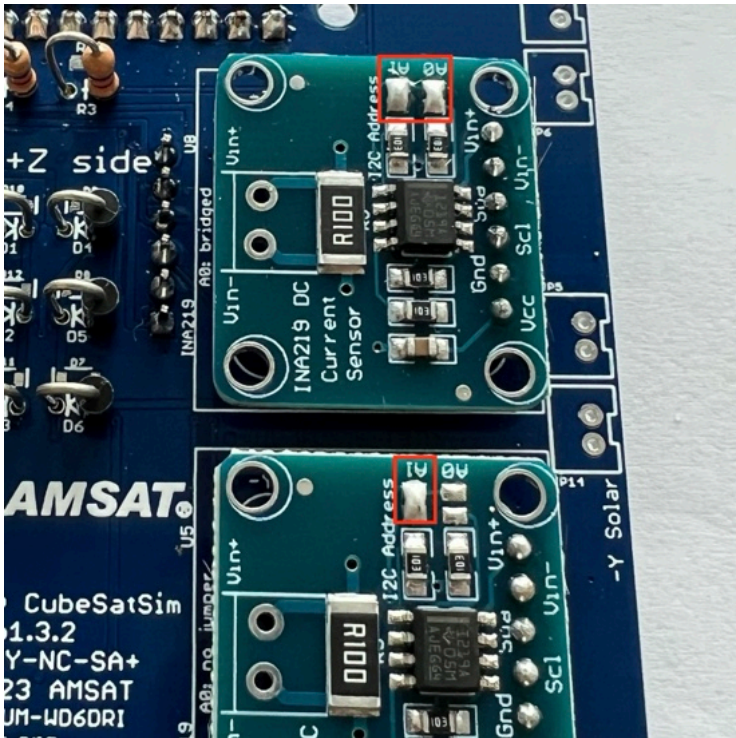
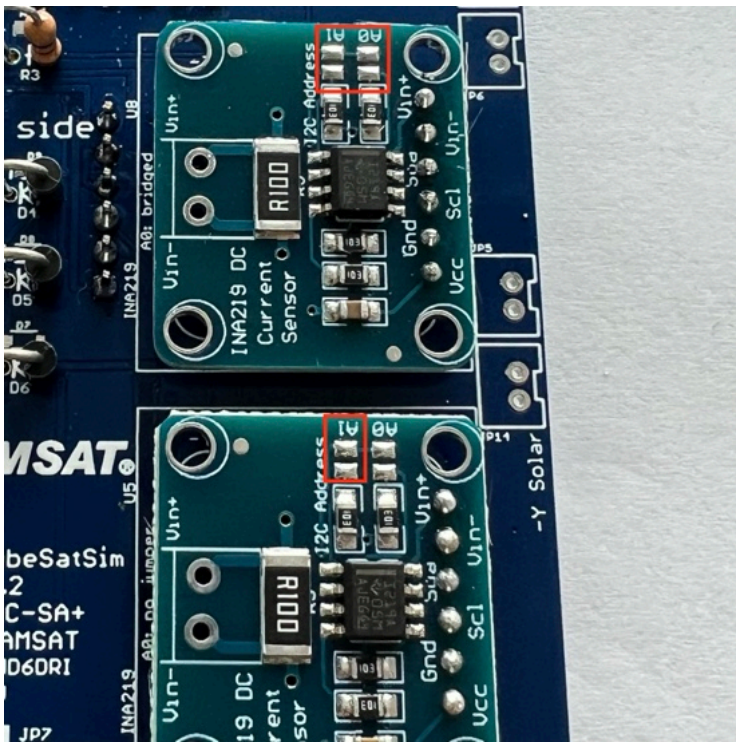


Solder one pin on each, check for straightness, then solder the rest:



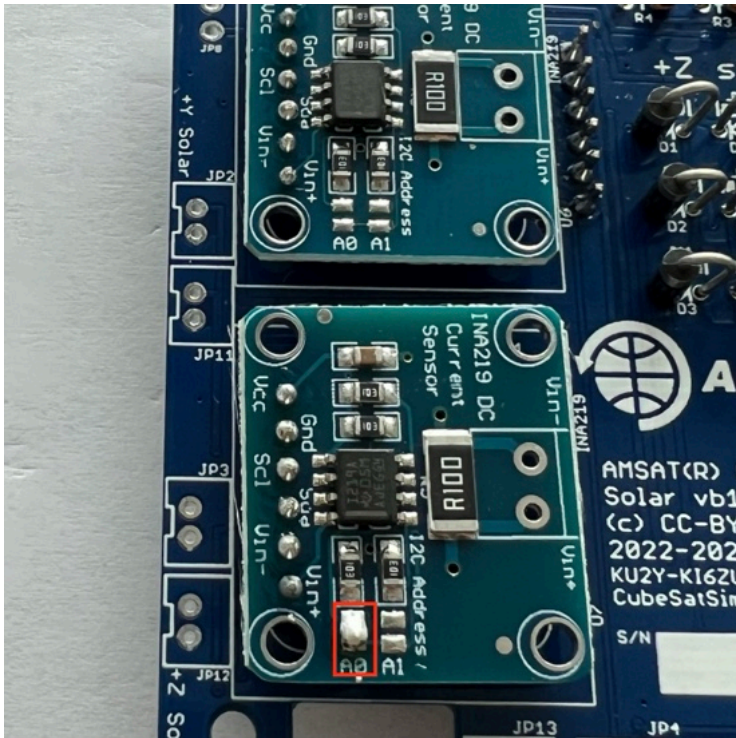
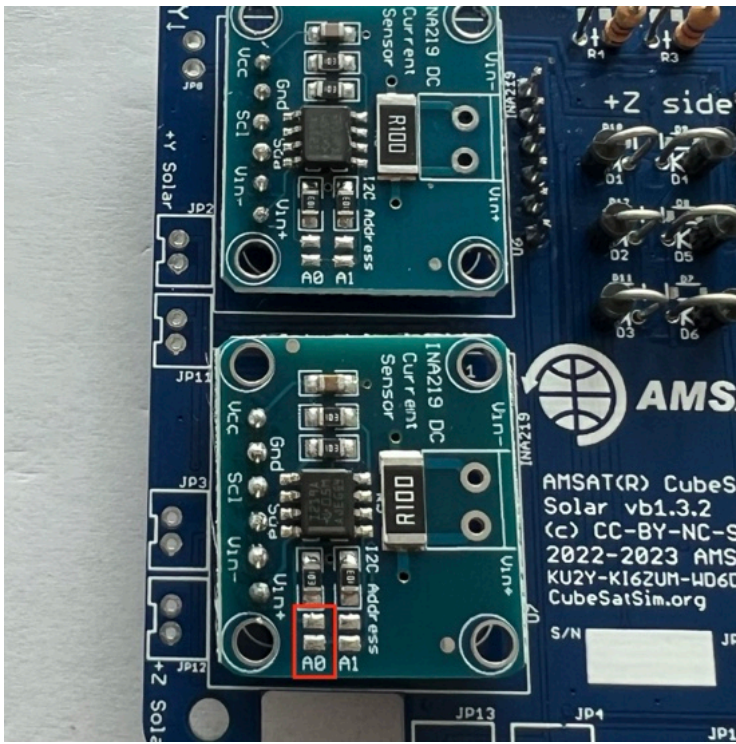
Next, the jumpers need to be set. On the right side, 3 jumpers need to be bridged:



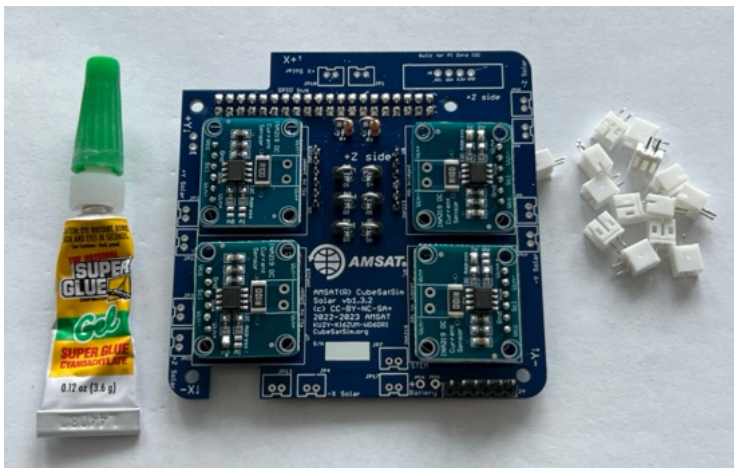


On the left side, one jumper needs to be bridged:

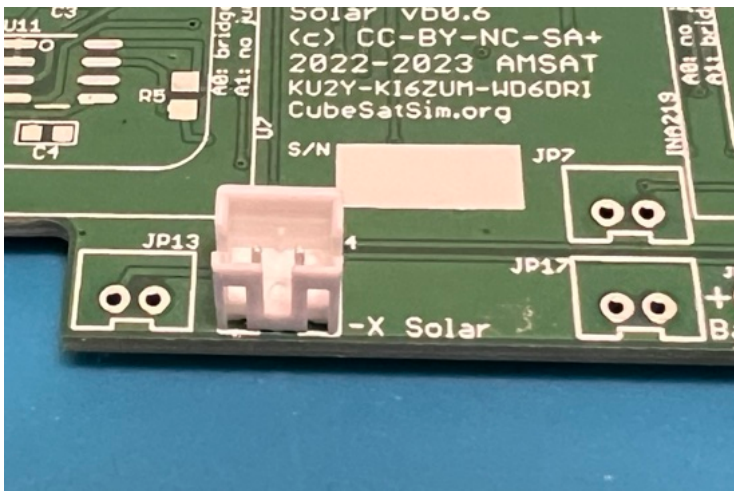




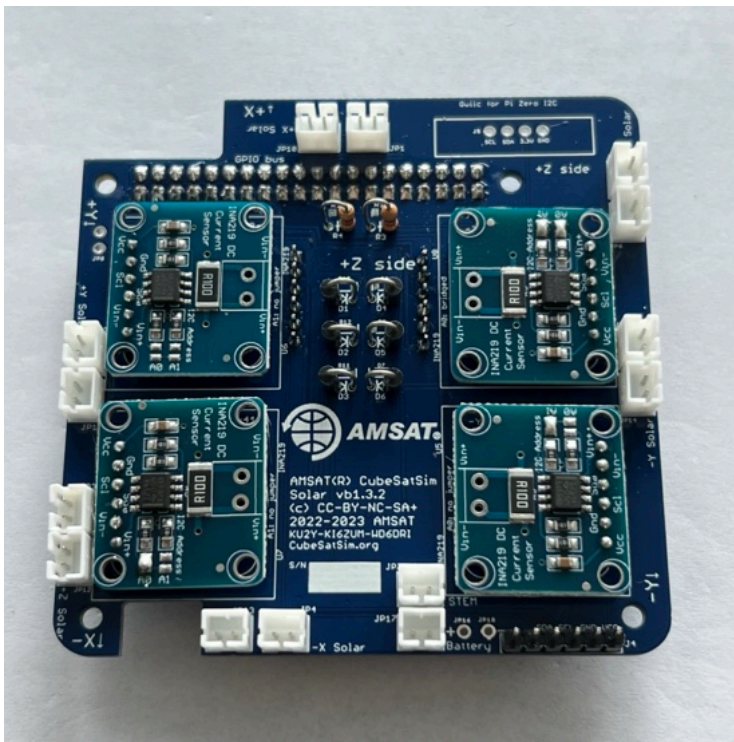
Next, install the JST connectors JP1, JP2, JP3, JP4, JP5, JP6, JP7, JP10, JP11, JP12, JP13, JP14, JP15, and JP17 on the top of the PCB. Some connectors click in place. If yours don't, you should add a drop of superglue gel or other adhesive to secure it to the PCB so it won't pop out when you are unplugging solar panels:



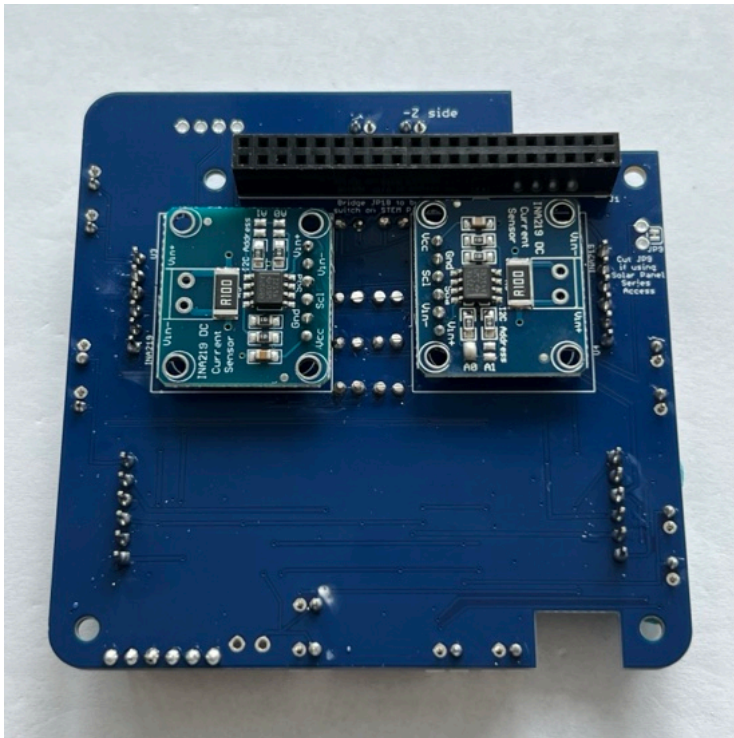
Note that these connectors have a polarity. Make sure the slot in the connector faces to the outside of the PCB:



Here's how the board looks with them all installed:

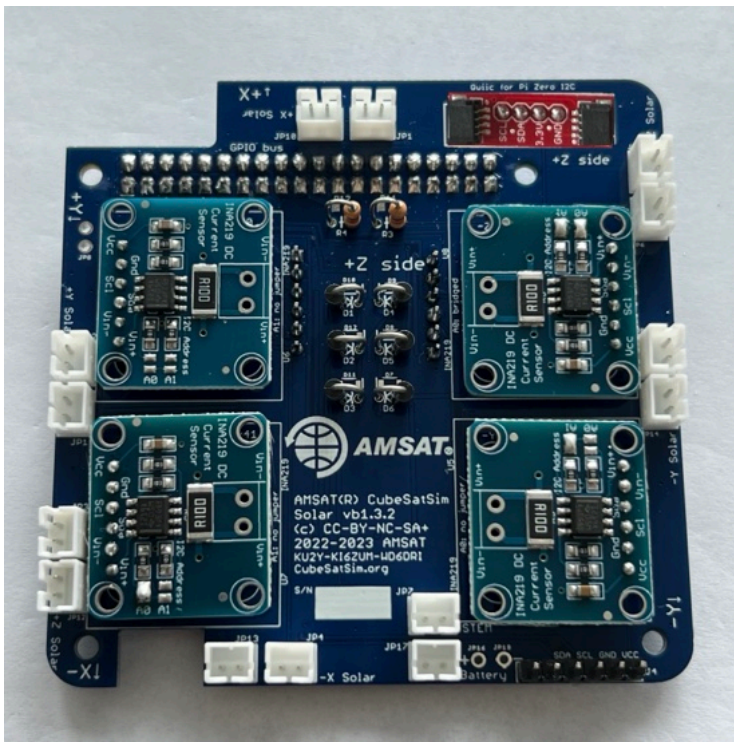


Flip the board over and solder one pin on each one then flip back and check insertion, orientation, and straightness. Then solder the other pin:



Next, install the Qwiic connector (red board) using the 1x4 pin male header J5.



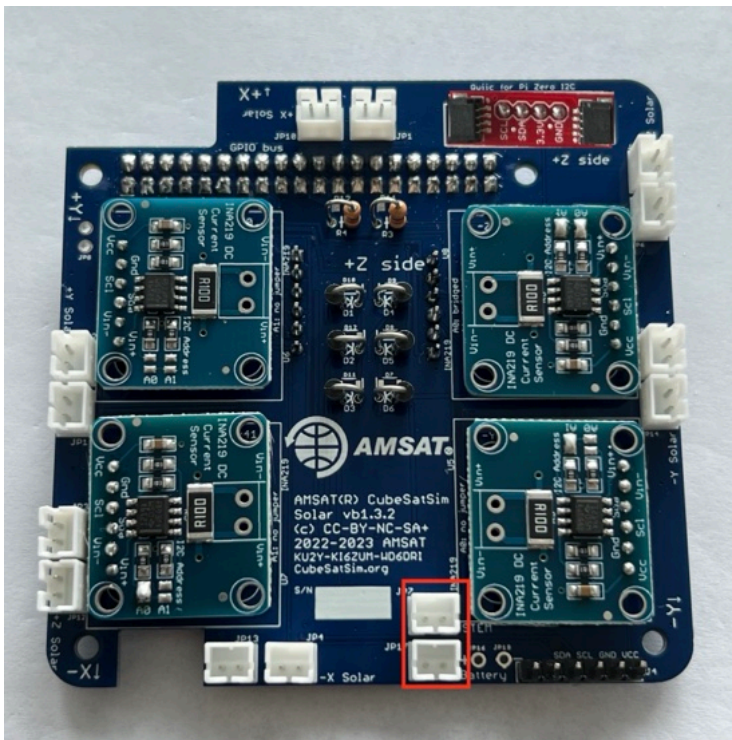


Here's the complete board:



The two JST in the red box are for the JST jumpers to the Battery and STEM Payload boards that you will connect in the Board Stack step. Both connectors are the same - it doesn't matter which is used.





Plug the JST cable into one of the connectors. You will plug in the other end into the STEM Payload board when you build the board stack.

If you have an old v1.3.2 STEM Payload PCB without the 2.5mm jack on it, you can add a 3.5mm plug to this PCB. You will make a 3.5mm plug out of a JST wire with these parts.

See [Command and Control information](#) for the details.

The next step is [Step 7. Solar Panels and Frame](#)

+ Add a custom footer

# 7. Solar Panels and Frame

Edit New page

Alan Johnston edited this page 3 weeks ago · [1 revision](#)

If the images on this page fail to load, you can [download a PDF of this page here](#).

## 7. Solar Panels and Camera

In this step, you will assemble and test the solar panels and plug the Pi Camera into the Pi Zero

### 7.1 Assembling and Testing the Solar Panels

These instructions are for the Solar Panels.

You will need these tools:

- Safety glasses (to protect eyes while soldering or trimming leads)
- Soldering iron and solder (I use lead-free solder, but leaded solder is easier to work with)

Other tools that are helpful:

- Multimeter (to read solar panel voltage)

### Video





- ▼ Pages 147
  - Find a page...
  - ▶ [Home](#)
  - ▶ [1. Main Board 1](#)
  - ▶ [2. Software Install](#)
  - ▶ [3. Ground Station](#)
  - ▶ [4. Main Board 2](#)
  - ▶ [5. Battery Board](#)
  - ▶ [6. Solar Board](#)
  - ▼ [7. Solar Panels and Frame](#)
    - 7. Solar Panels and Camera
      - 7.1 Assembling and Testing the Solar Panels
        - Video
        - Checklist
        - Solar Panels
          - Video
          - Solar Panel Soldering.
          - Solar Panel Testing
            - Open Circuit Testing
            - Short Circuit Testing
      - 7.2 Pi Camera Instructions

Here is a [video of this step](#)

## Checklist

The BOM has a sheet "By Steps" which lists the parts needed for each step in order. If you have a Google account, you can make a copy of this spreadsheet ("File" then "Make a Copy") and check off each part as you install it.

For example, here is the checklist for this step:

Step 7. Solar Panels and Camera		<a href="https://github.com/alanbjohnston">https://github.com/alanbjohnston</a>			
<input checked="" type="checkbox"/>	Item	Qty	Location	Image	
<input type="checkbox"/>	Solar Cell 6V 60mA 72mm x 45mm	10			
<input type="checkbox"/>	Micro JST wires	10			
<input type="checkbox"/>	Pi Zero with SD card from Step 2	1			
<input type="checkbox"/>	Pi Camera with Pi Zero ribbon cable	1			

## Solar Panels

These instructions are for soldering, testing, and trial mounting on the space frame.

## Video

Here is a video of this step.

Any solar panel that is 5V - 6V and less than 90mm (45mm if two panels are used) in length will work. For side with the camera and the side with the pushbutton, USB-C charging port, and the LEDs, the longest dimension of the solar panel is 45mm.

### Video

- ▶ [8. Board Stack](#)
- ▶ [9. Final Testing](#)
- ▶ [Adding New Sensors](#)
- ▶ [Command and Control](#)
- ▶ [Creating the CubeSatSim Raspberr...](#)
- ▶ [CubeSatSim Lite](#)
- ▶ [CubeSatSim Loaner User Guide](#)

Show 132 more pages...

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/alanbjohnston>



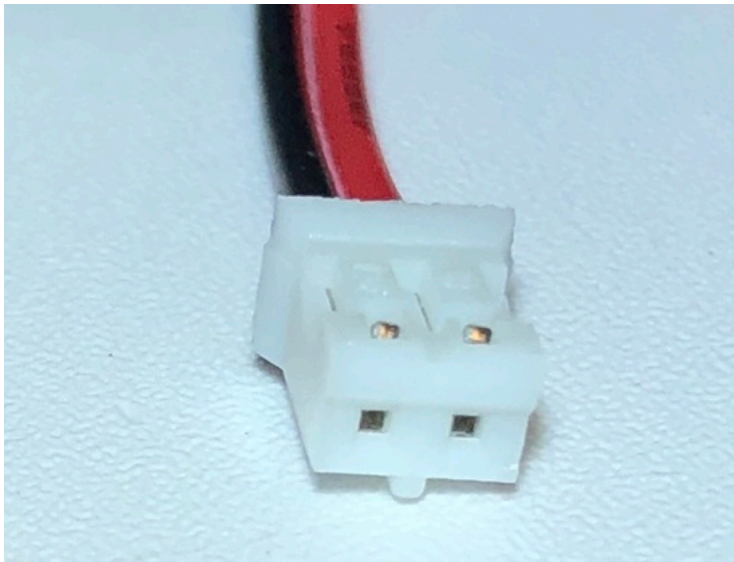
The recommended solar panels are 72mm x 45mm, and there are two mounted on each side, except the top and bottom that have just one. (You can also mount two solar panels on the top and bottom, although the solar panels are commonly sold in batches of 10.

Your set of solar panels may contain a mix of panels.

## Solar Panel Soldering.

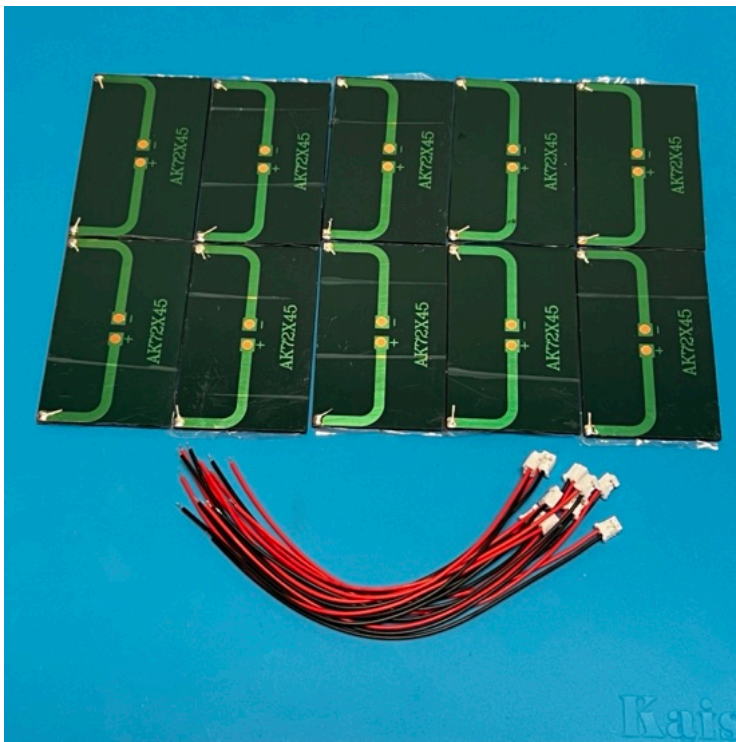
---

Before using any of your JST connector wires, make sure they have the correct polarity. There is no standard in the industry for red and black, unfortunately, and some suppliers will supply different polarities in different orders. Verify your polarity against this image, and swap if they are reversed:

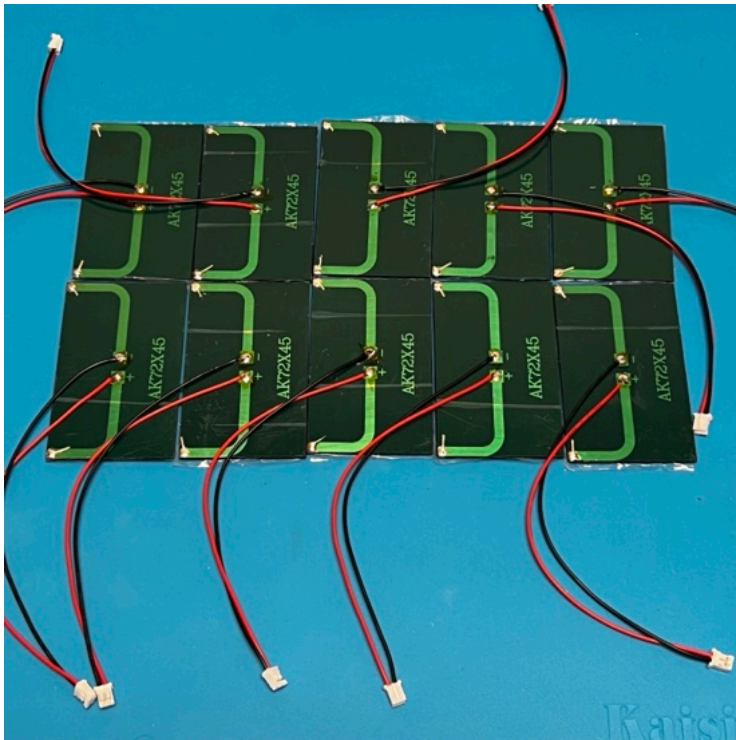


The 10 solar panels need to have the JST 2.0 connectors soldered on:

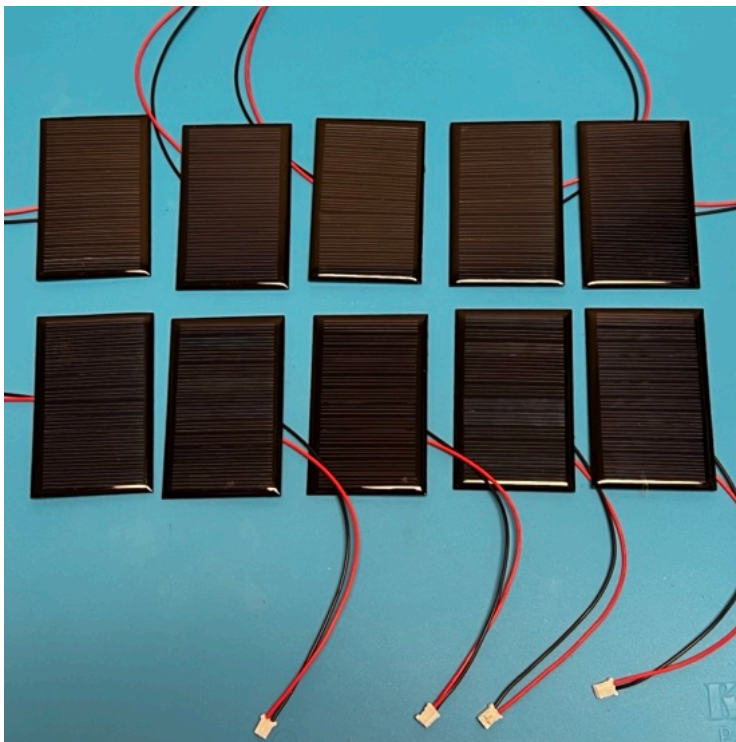




Some liquid flux can be applied to each pad of the solar panels to make soldering easier. Also, hot glue can be used to stick the insulation of the wires to the panel to provide extra strength.



Here's how the panels look after the protective film is removed:



## Solar Panel Testing

---

The solar panels are tested with the Digital Multi Meter (DMM). They should be tested in constant illumination such as from the LED lamp.

### Open Circuit Testing

Open circuit testing measures the maximum voltage output. Put the DMM on the DC volts scale and measure while illuminated.

Test each panel to make sure they work.

Note down the maximum voltage for each size of solar panel.

### Short Circuit Testing

Short circuit testing measures the maximum current output from the solar panel. Put the DMM in DC Current (or Amps) mode and measure across the red and black leads while illuminated.

Test each panel to make sure they work.

Note down the maximum voltage for each size of solar panel.

The Solar Panels are now ready to attach to the frame.

## 7.2 Pi Camera Instructions

---

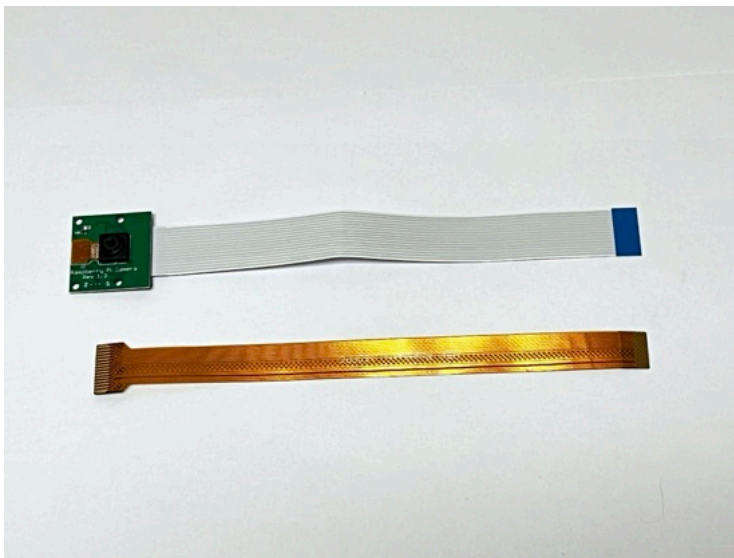
Here are the parts you need for the Pi Camera:

- Side frame with camera mount
- Two small 80mm x 35mm solar panels (or 57mm x 28mm), wired in parallel
- Pi Camera
- 6.3 inch (16 mm) Camera cable for the Pi Zero W

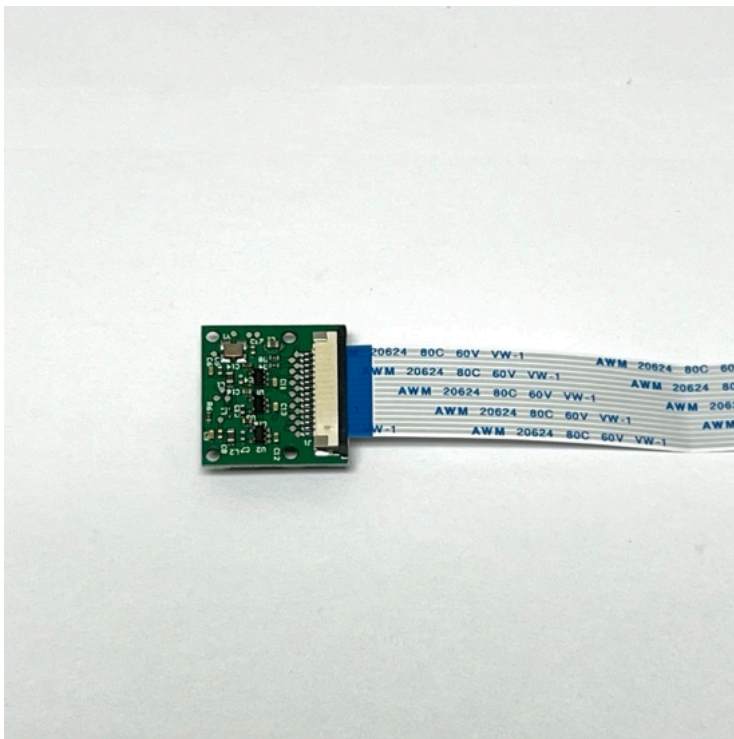
### Video

Here is a video on the Raspberry Pi Camera Installation (only the first 8 minutes are relevant): <https://youtu.be/5ras2Y0Cfec>

Most Pi Cameras come with the cable to connect to a full sized Pi, so you may need to buy a Pi Zero W Camera Cable as well:

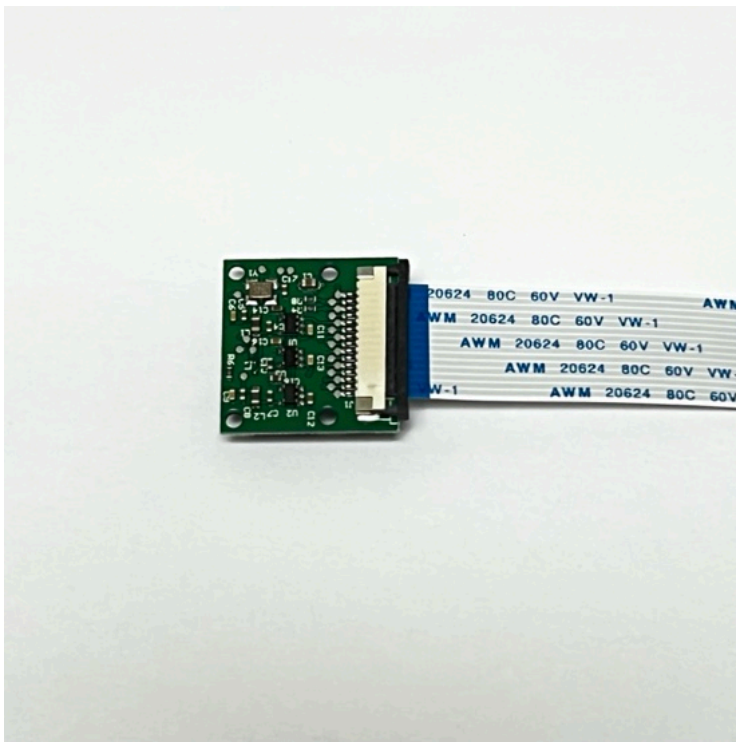


To disconnect the Pi Camera Cable, identify the slide lock on the connector on the back of the Pi Camera. In this photo it is the black plastic piece:

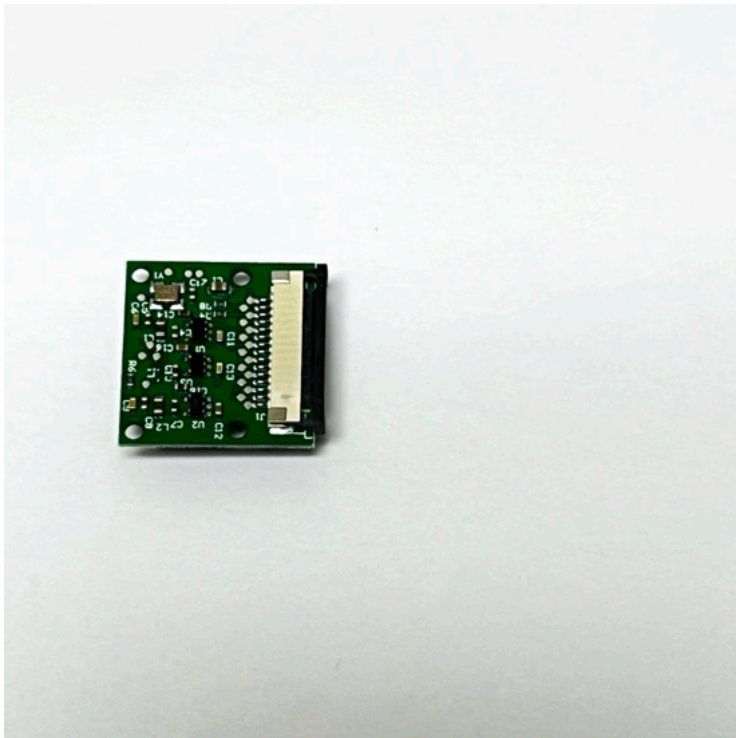


Carefully move the slide lock away from the board, to the right in this photo. You might need to move it a little on one side, then the other side. It only moves out a few millimeters and does not come completely off:

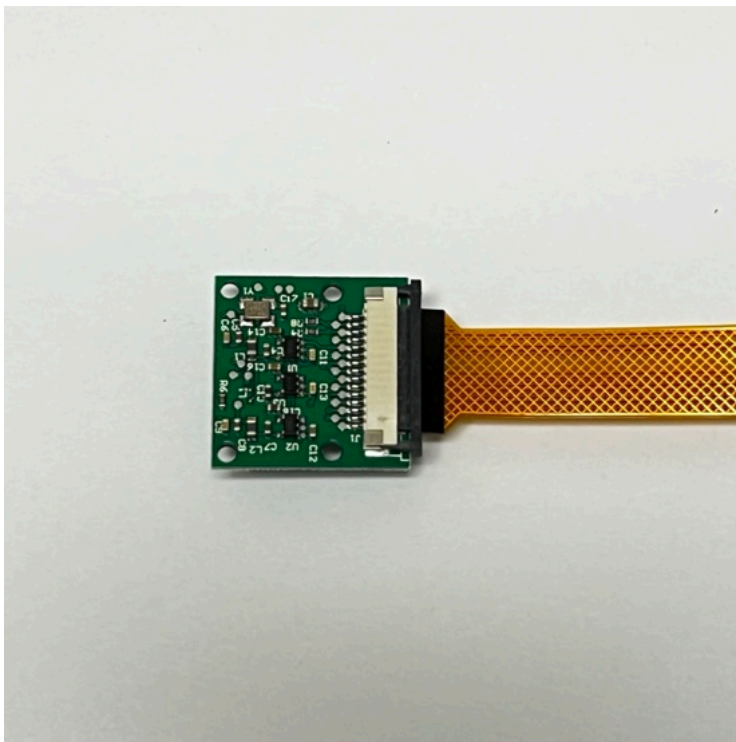




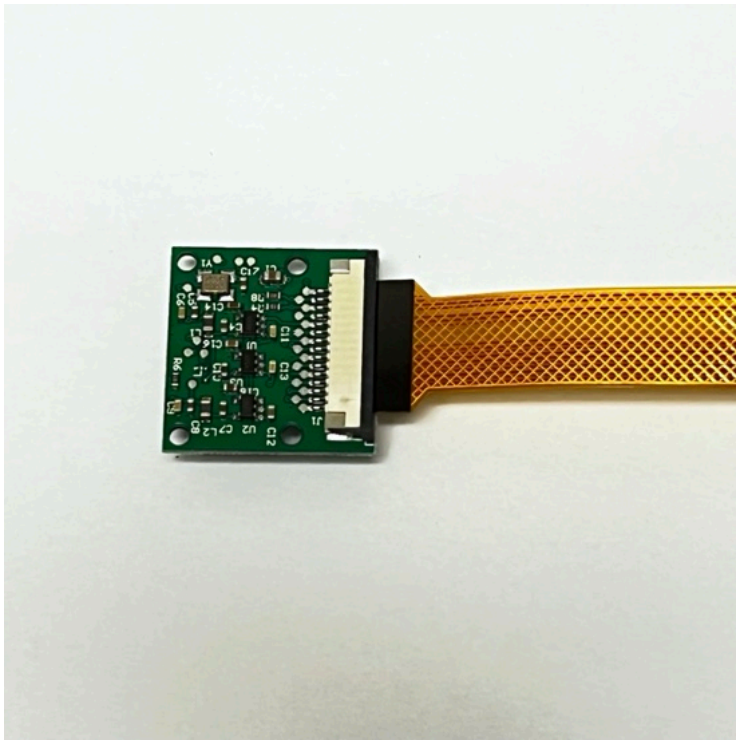
The cable will now be loose and you can slide it out and remove it.



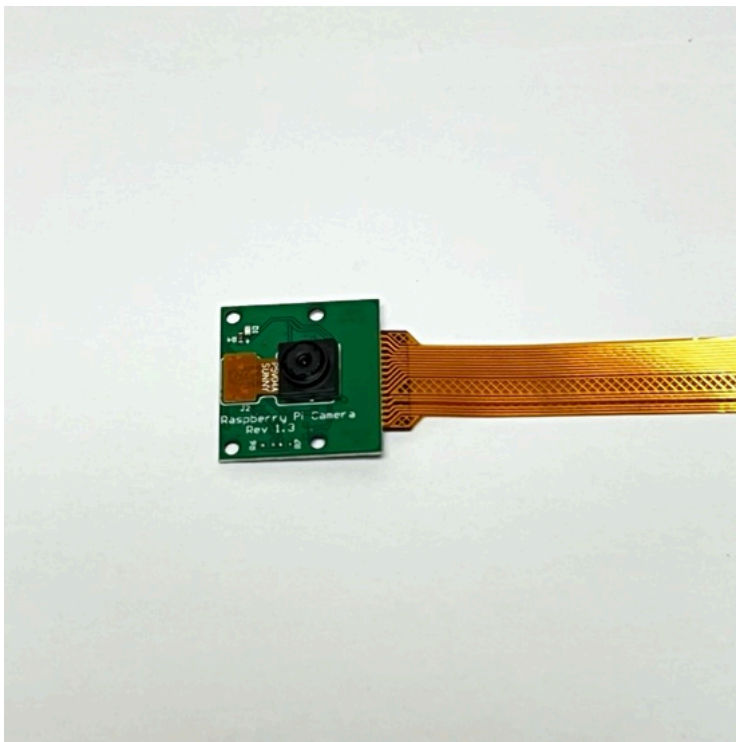
Slide the 6.3 inch (16 mm) Pi Zero Camera Cable into the connector. Make sure the metal contacts on the cable are facing the PCB, in this photo the contacts face down:



Carefully move the slide lock towards the board (to the right in this photo) to hold the cable securely. Make sure the cable doesn't slide over or get crooked:

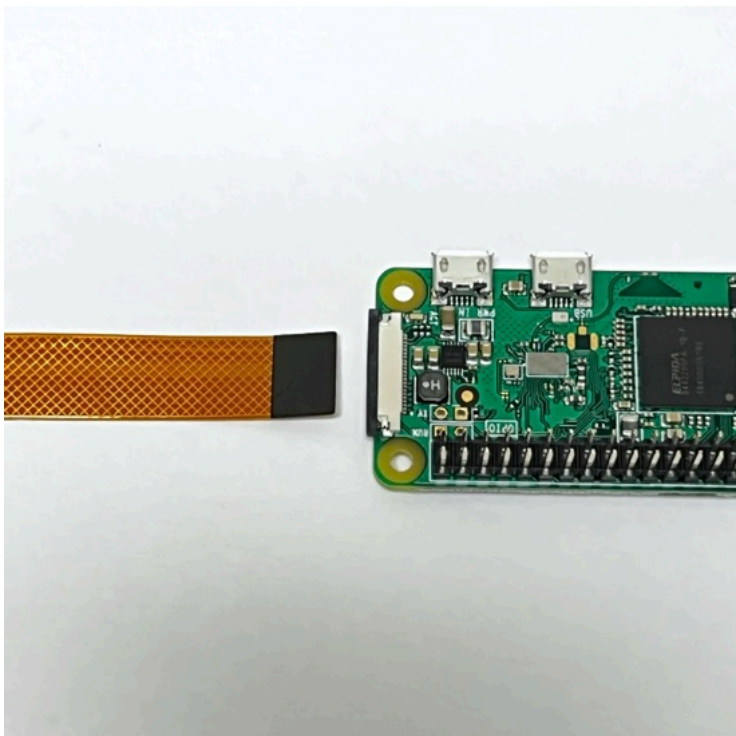


This is how it looks on the other side.

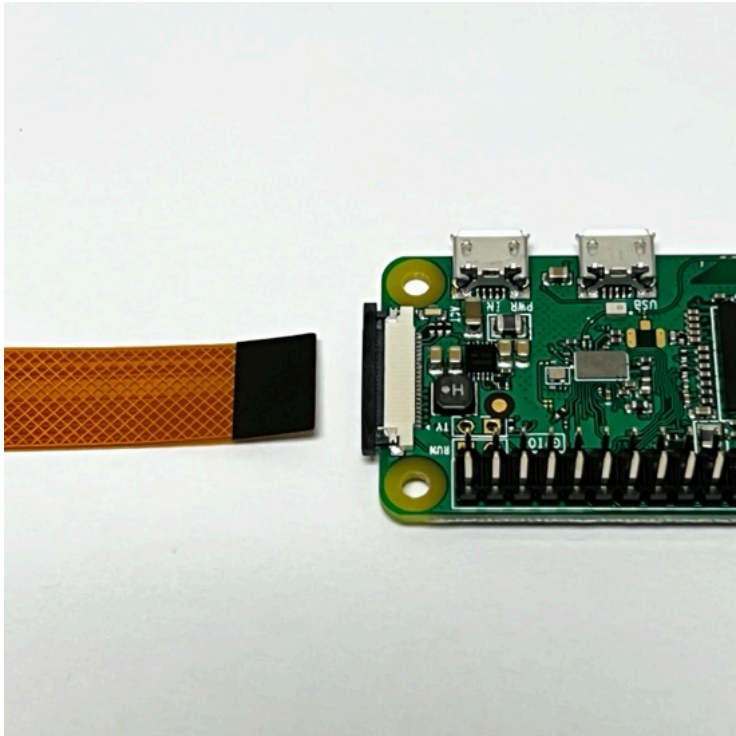


Don't forget to remove the plastic film over the camera lens!

The camera cable plugs into the Pi Zero on the opposite side to the micro SD card. This is how the Pi Zero Camera Cable and the Pi look when ready to connect:

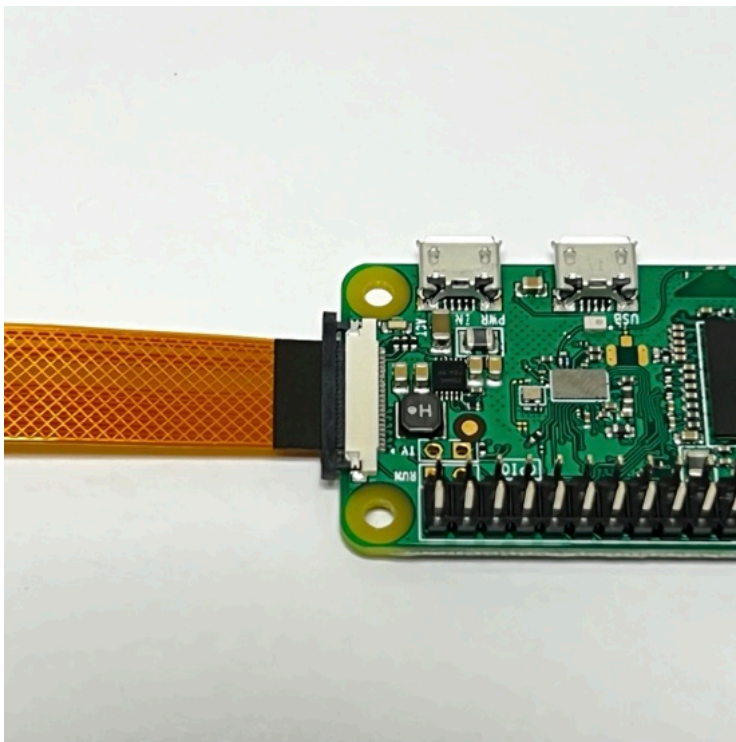


The other end of the Pi Zero Camera Cable connects to the Pi Zero WH on the opposite side to the micro SD card slot. Identify the slide lock, in this photo it is the black plastic:

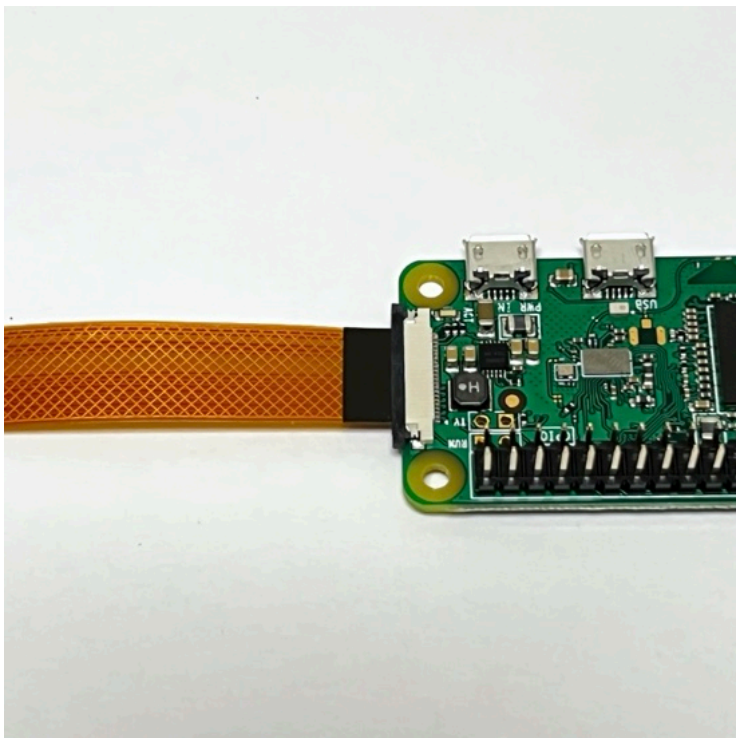


Carefully move the slide lock away from the board, in this photo to the left. Be very very careful - it is very easy to break it. You might need to move it a little on one side, then the other side. It only moves out a few millimeters and does not come completely off. There may be a plastic "blank" in the slot - you can remove it.



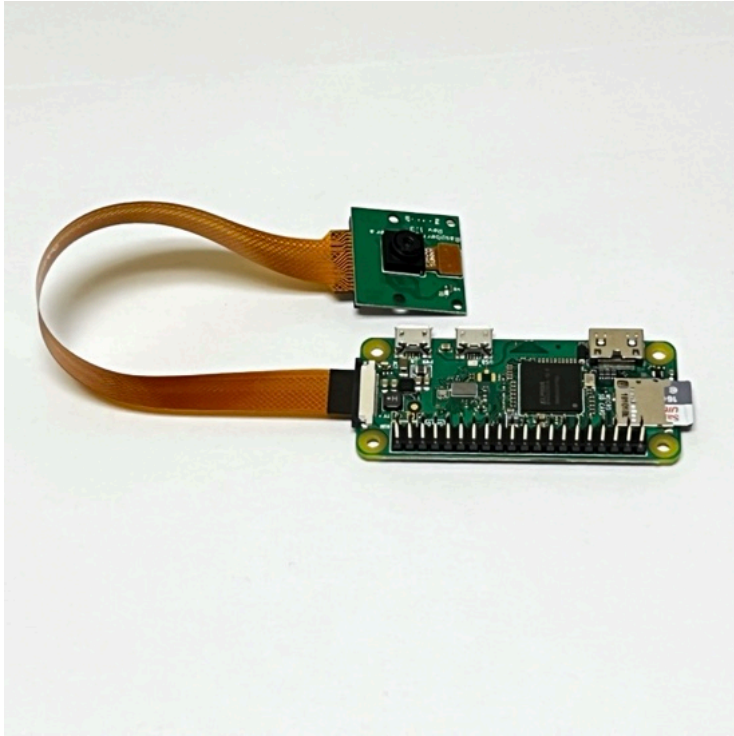


Insert the Pi Zero Camera Cable into the connector. Make sure the metal contacts are facing the PCB, in this photo facing down:



Carefully move the slide lock towards the board (to the left in this photo) to hold the cable securely. Make sure the cable doesn't slide over or get crooked.

Here is what the Pi Zero W and the Pi Camera look like:



Note that the LED on the Pi Camera will blink once when the Pi Zero boots up and whenever it takes a photo.

The next step is to [Step 8 Board Stack](#).

+ Add a custom footer

# 8. Board Stack

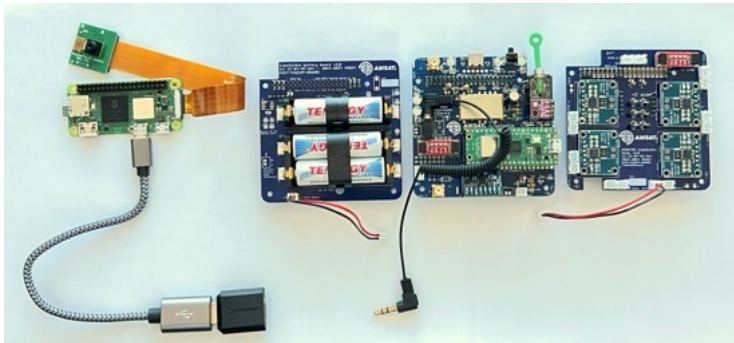
Edit New page

Alan Johnston edited this page 5 hours ago · [21 revisions](#)

If the images on this page fail to load, you can [download a PDF of this page here](#).

## 8. Board Stack

The four boards in the board stack are shown here:



You will need these tools:

- Small Philips screwdriver
- Scissors to cut double stick tape or velcro
- Small flathead screwdriver (only if you are doing the tape measure antenna)

## Checklist





















▼ Pages 147

Find a page...

- ▶ [Home](#)
- ▶ [1. Main Board 1](#)
- ▶ [2. Software Install](#)
- ▶ [3. Ground Station](#)
- ▶ [4. Main Board 2](#)
- ▶ [5. Battery Board](#)
- ▶ [6. Solar Board](#)
- ▶ [7. Solar Panels and Frame](#)
- ▼ [8. Board Stack](#)
  - 8. Board Stack
    - Checklist
    - 8.1 Stacking the Boards
      - Video
    - 8.2 Installing the Board Stack into the Frame
      - Printing the CubeSatSim Frame
        - Frame Print
      - Frame Test Assembly with Solar Panels

The BOM has a sheet "By Steps" which lists the parts needed for each step in order. If you have a Google account, you can make a copy of this spreadsheet ("File" then "Make a Copy") and check off each part as you install it.

Here is the checklist for this step:

		Step 8. Board Stack and Frame	<a href="https://github.com/alanbjohnston">https://github.com/alanbjohnston</a>			
<input checked="" type="checkbox"/>	Ref	Item	Qty	Location	Image	
<input type="checkbox"/>		Pi Zero with SD card and Pi Camera	<input type="checkbox"/>	1		
<input type="checkbox"/>		Battery Board fully assembled	<input type="checkbox"/>	1		
<input type="checkbox"/>		Main Board fully assembled	<input type="checkbox"/>	1		
<input type="checkbox"/>		Solar Board fully assembled	<input type="checkbox"/>	1		
<input type="checkbox"/>		GPIO 20x2 female stacking header extra long		2		
<input type="checkbox"/>		M2.5 screws		8		
<input type="checkbox"/>		M2.5 23mm + 6mm standoff		8		
<input type="checkbox"/>		M2.5 11mm standoff		2		
<input type="checkbox"/>		M2.5 18mm standoff		2		
<input type="checkbox"/>		M2.5 6mm + 6mm standoff		2		
<input type="checkbox"/>		USB Sound Card		1		
<input type="checkbox"/>		OTG cable for Sound Card		1		
<input type="checkbox"/>		2.5mm to 3.5mm jumper cable		1		
<input type="checkbox"/>		3D Printed Frame (4 parts)		1		
<input type="checkbox"/>		Nylon M3 screws for frame		10		
<input type="checkbox"/>		Nylon M3 nuts for frame		10		
<input type="checkbox"/>		Slotted nylon M2 screws for camera		4		
<input type="checkbox"/>		nylon M2 nuts for camera		4		
<input type="checkbox"/>		Solar Cells from Step 7		10		
<input type="checkbox"/>		Velcro thin clear fasteners		1		

Video

- ▶ [9. Final Testing](#)
- ▶ [Adding New Sensors](#)
- ▶ [Command and Control](#)
- ▶ [Creating the CubeSatSim Raspber...](#)
- ▶ [CubeSatSim Lite](#)
- ▶ [CubeSatSim Loaner User Guide](#)

Show 132 more pages...

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/alanbjohnston>





# 8.1 Stacking the Boards

---

This step will stack the three boards and the Pi Zero.

## Video

---

Here is a [video of stacking the boards](#)

In order to stack the boards, we will need a set of 2.5mm steel, brass, or nylon spacers and two GPIO stacking headers.

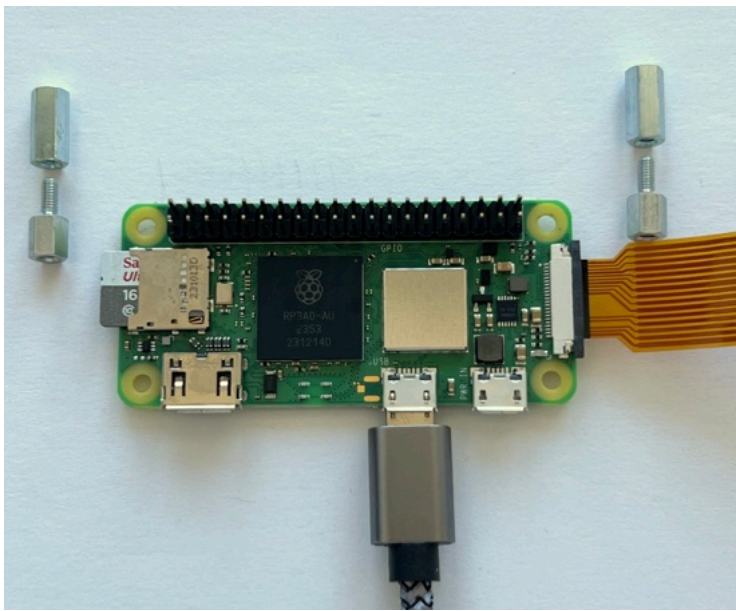
Standoffs are described by the thread size (M2.5 in this case), length of the standoff (6, 10, and 15mm), and length of the standoff threads if any (+6mm). Also M2.5 screws and nuts are used. Here are the standoff pieces needed for the three board stack along with their names in their correct order and orientation. On the left is if you order the individual pieces using the BOM. On the right is if you buy a standoff set such as <https://www.amazon.com/gp/product/B08HS7MFYZ/> and combine them. These instructions will describe the BOM parts with the kit combined parts in parentheses. The photos may show the combined parts.



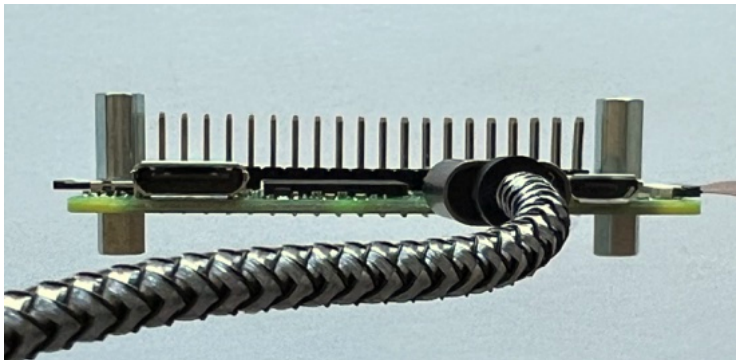
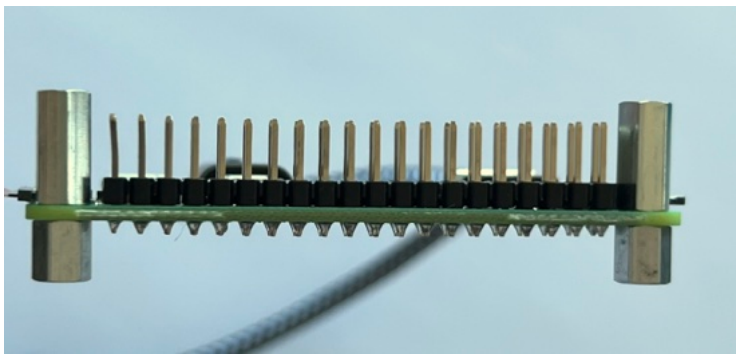
If you use the standoff kit, you will need 8x M2.5 screws, 10x M2.5 nuts, 12x 6+6mm, 8x 15+6mm, and 4x 10mm.

You will also need two stacking GPIO headers to give extra spacing between the boards. They are identical to the ones soldered onto the Battery, Main, and Solar boards.

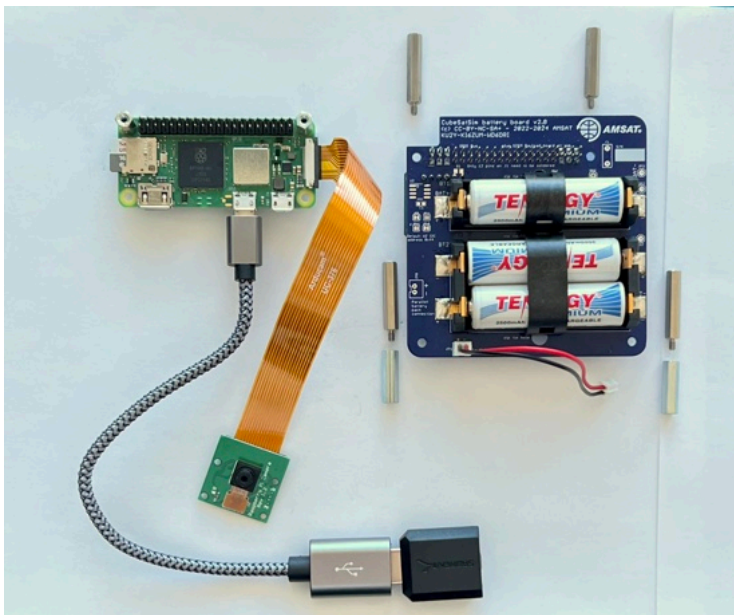
Start with the Pi Zero 2 and two 11mm and 6+6mm standoffs (or the 10mm and 6+6mm standoffs):



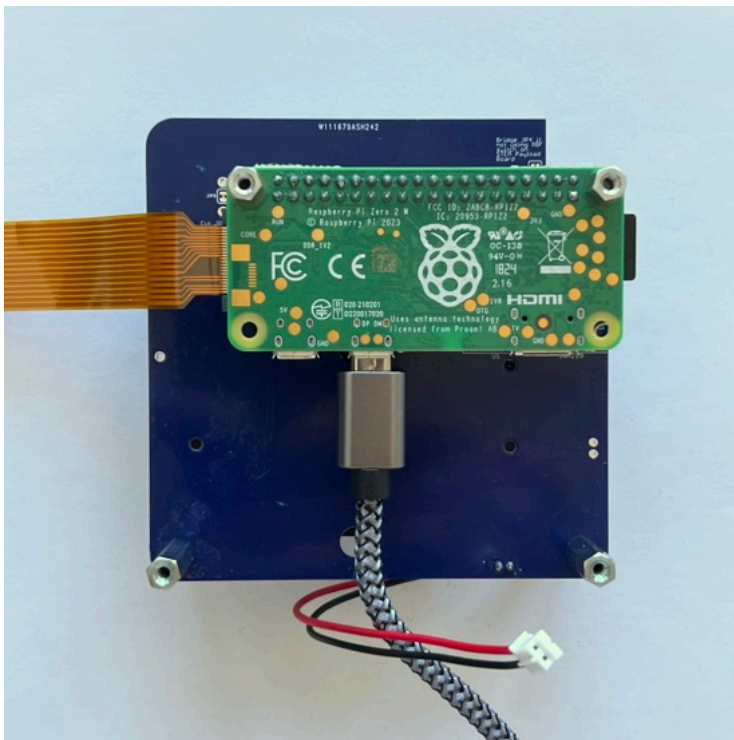
The 11mm goes on top, the 6+6mm goes on the bottom. They go in the holes closest to the GPIO header pins



Next, take the Battery board. You will use 4x 23+6mm standoffs, and 2x 18mm standoffs (from the kit, make two sets of 10mm and a nut and a 6+6mm, and four sets of 6+6mm and a nut and a 15+6mm standoff).

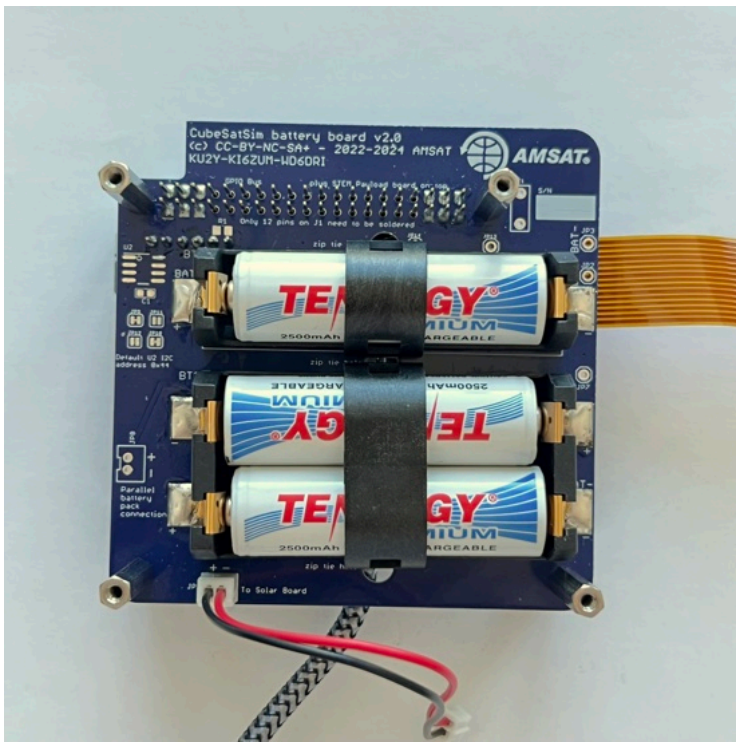


Plug the Pi Zero into the bottom of the Battery board. The two 18mm standoffs (10mm, nut, and 6+6mm standoffs) will go under the other two corners of the Battery board

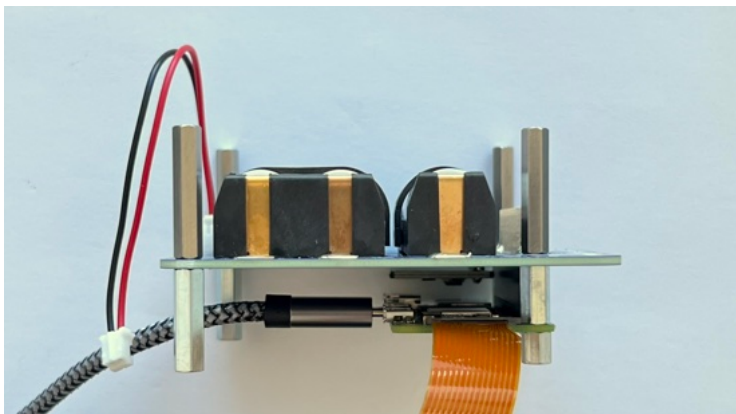
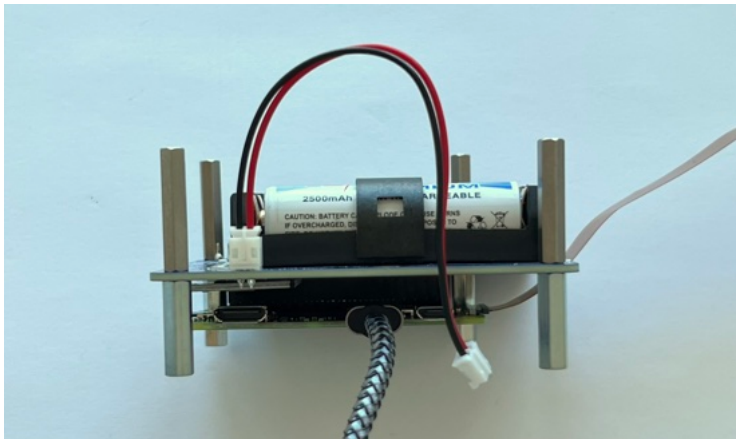


Screw the four 23+6mm standoffs (6+6mm, nut, and 15+6mm standoffs) into the top of the Battery board

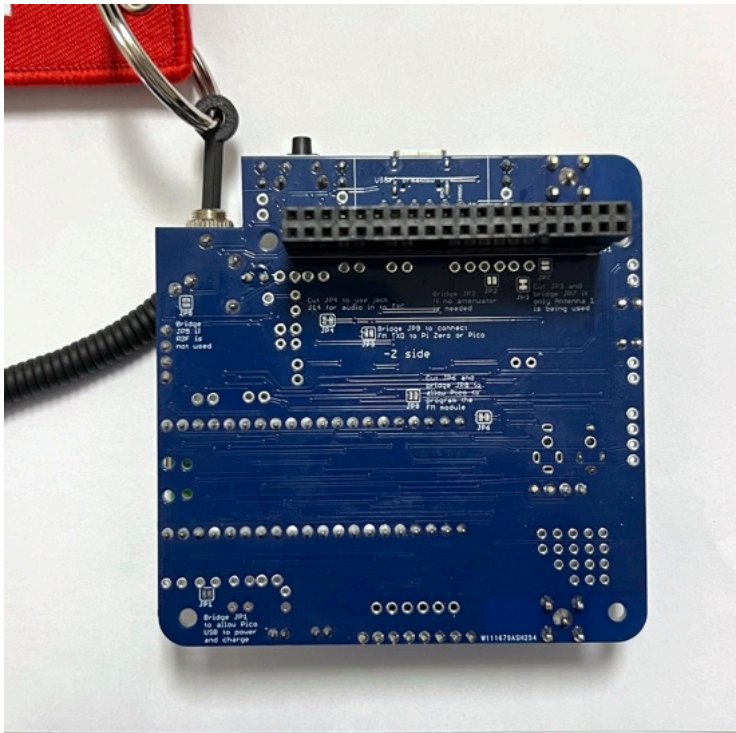
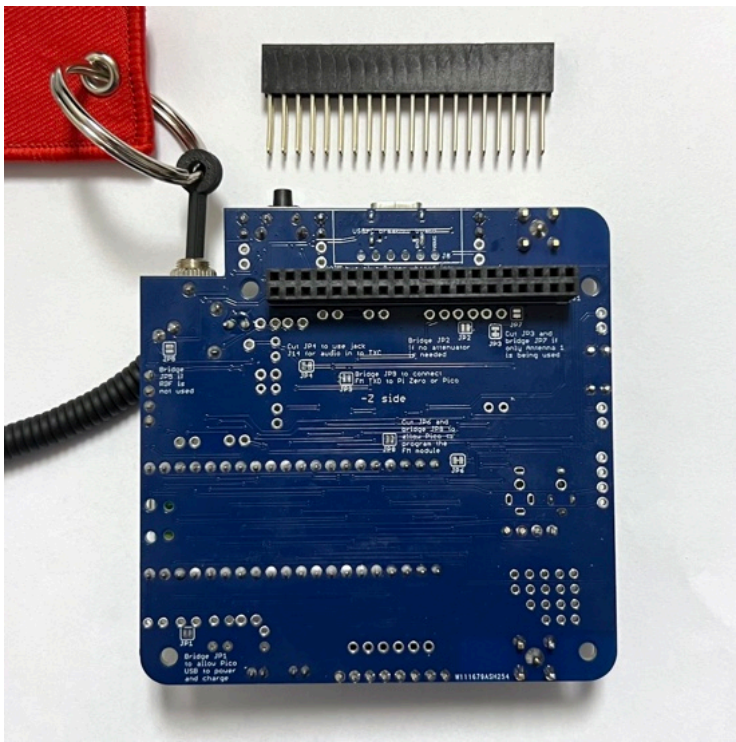


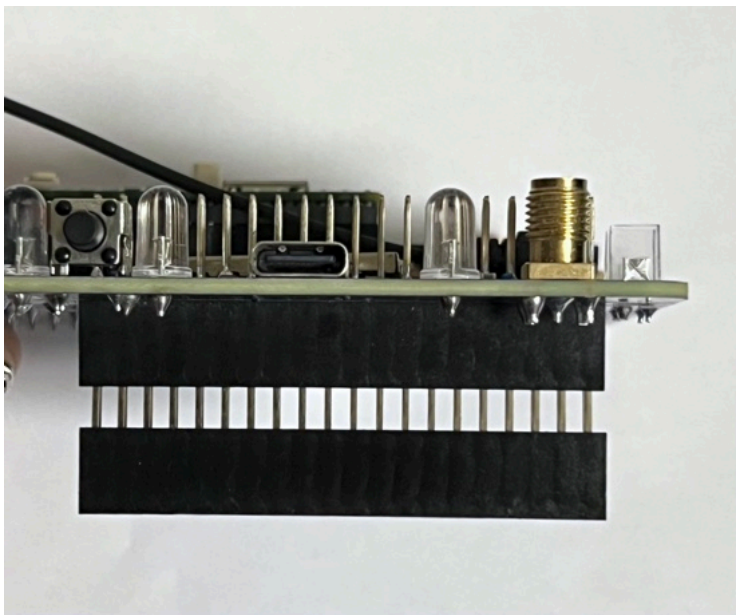


Here's how it looks from the sides:

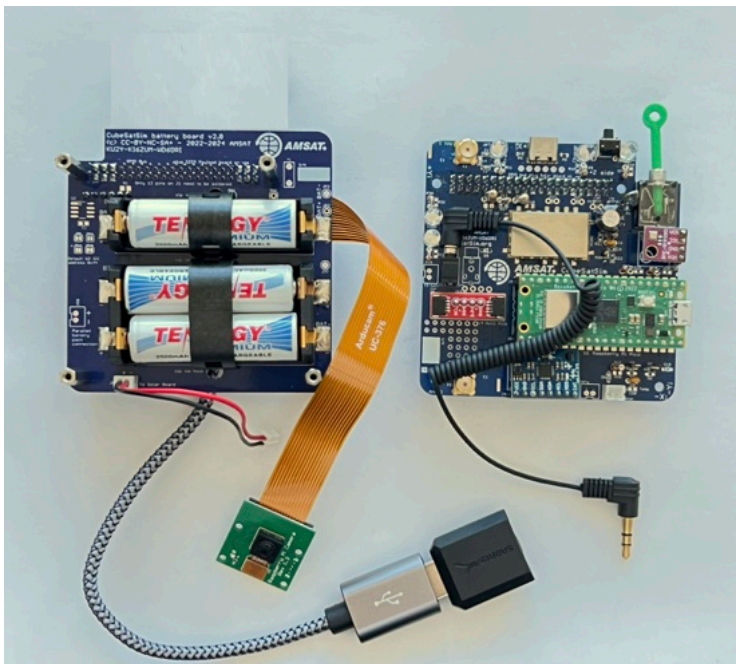


Next, put a stacking GPIO header onto the bottom of the Main board



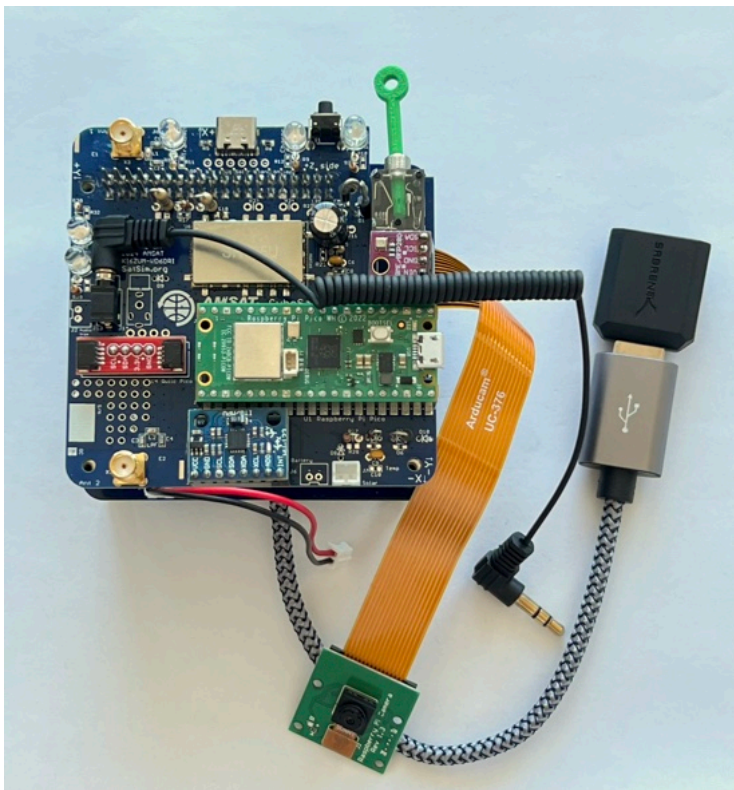


The Main board is then stacked on top of the Battery Board:

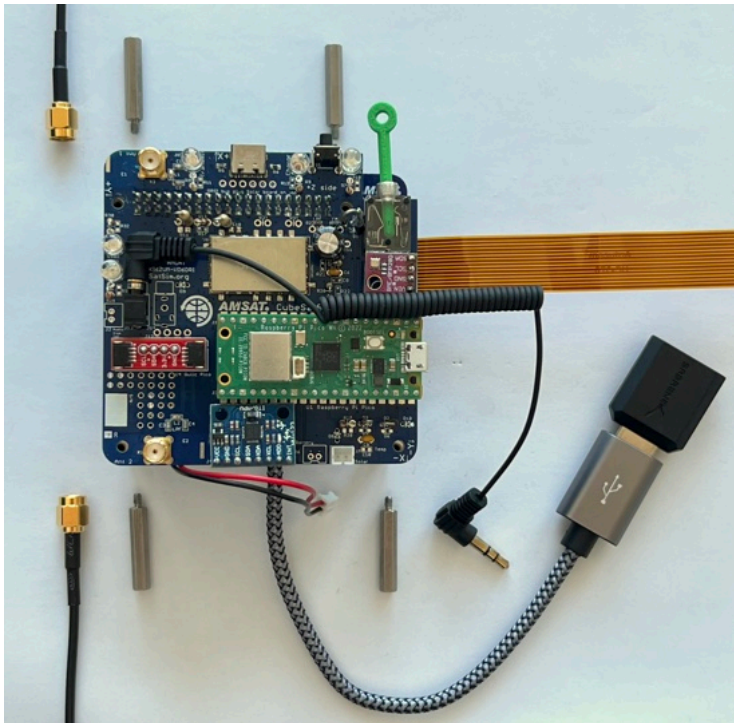


Here's how it looks:



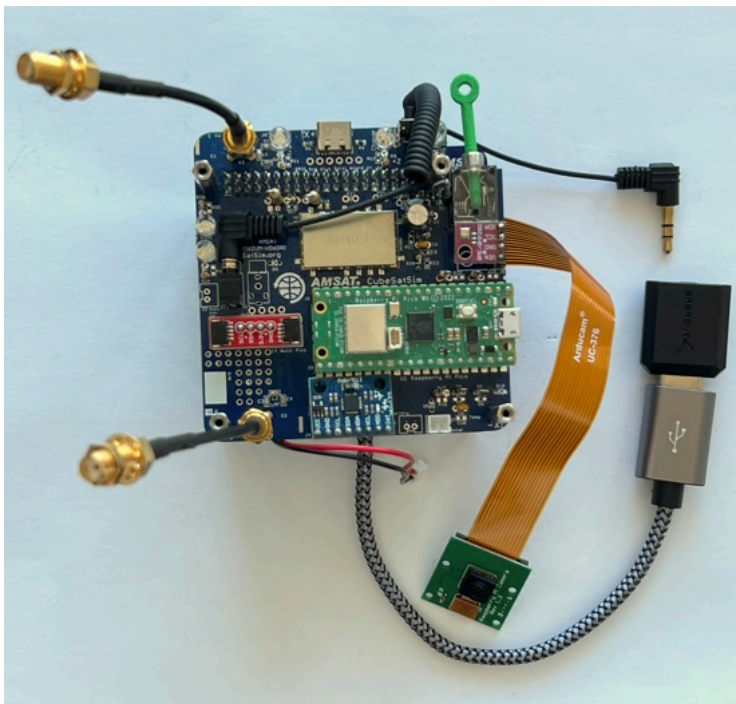


Next, use four 23+6mm standoffs (make up four more 6+6mm, nut, and 15+6mm standoffs) and screw them into the top of the Main board.



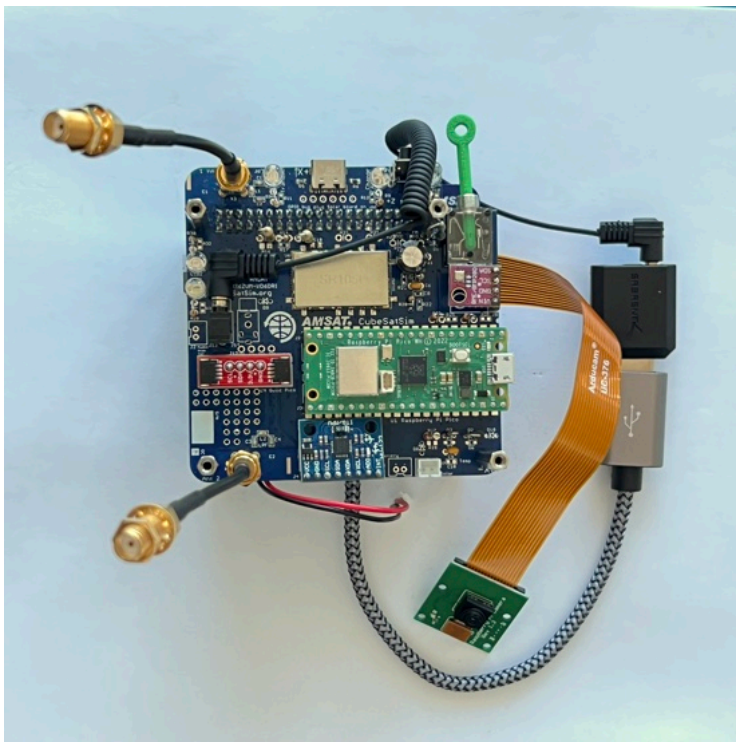
If the SMA antenna is in use, make sure the coax cables are screwed in.



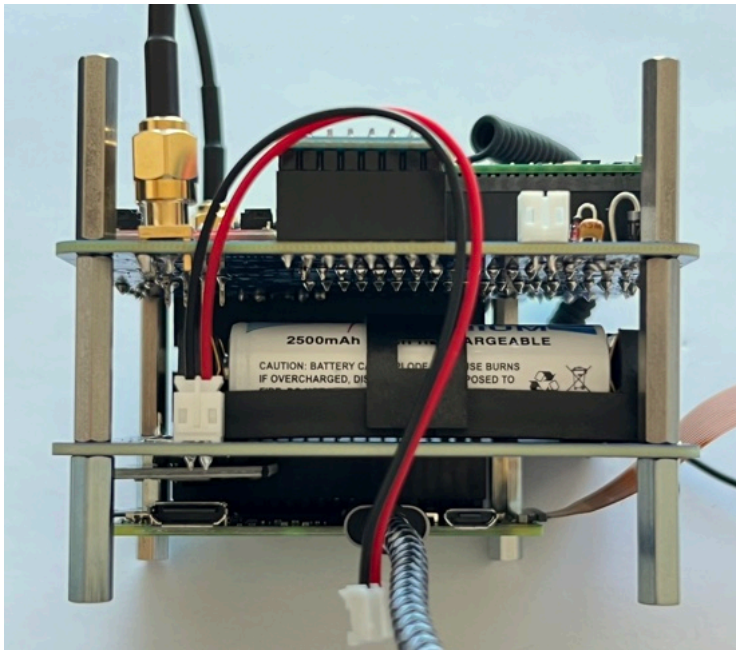


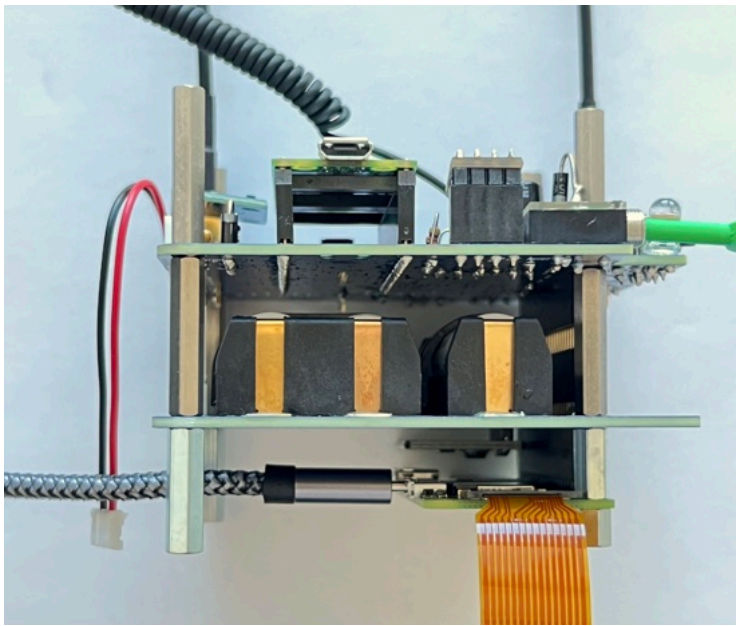
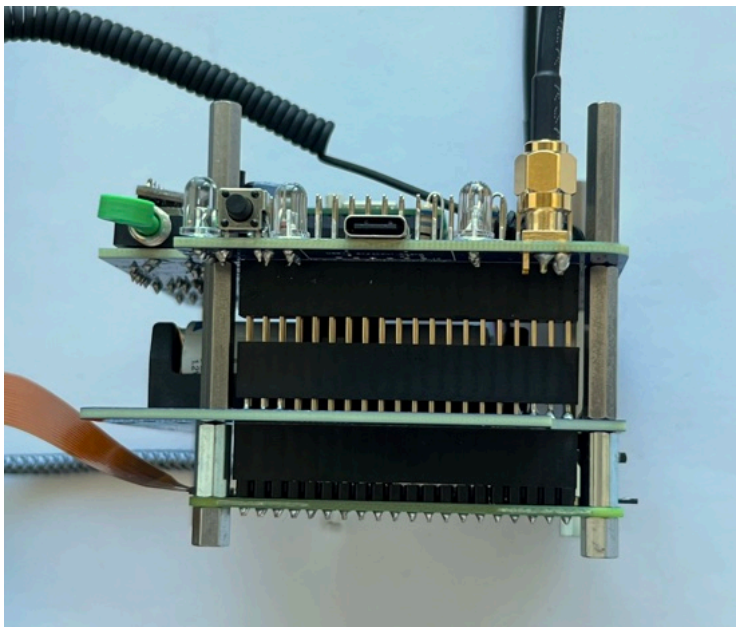
Plug the 3.5mm jumper from the Main Board into the microphone jack (pink jack) on the USB Sound Card:





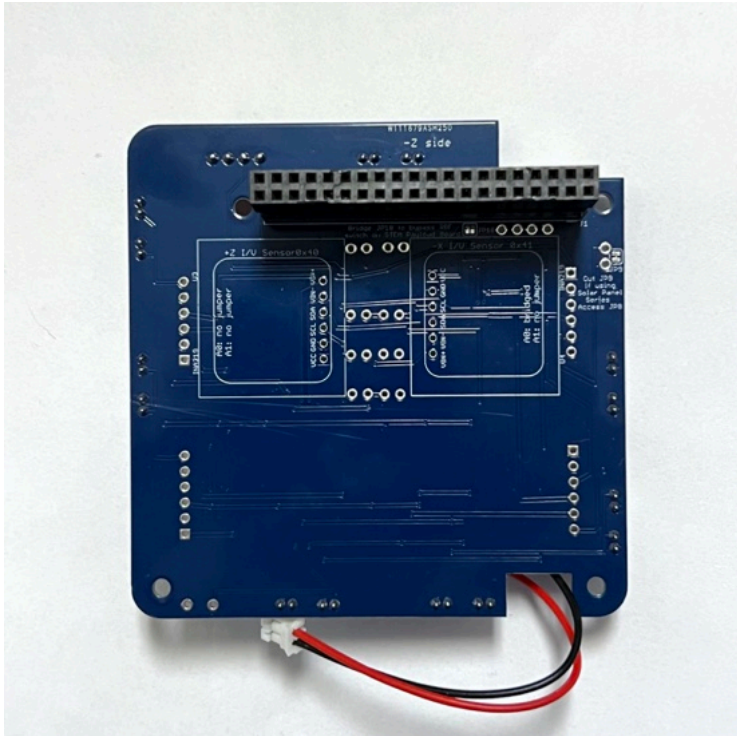
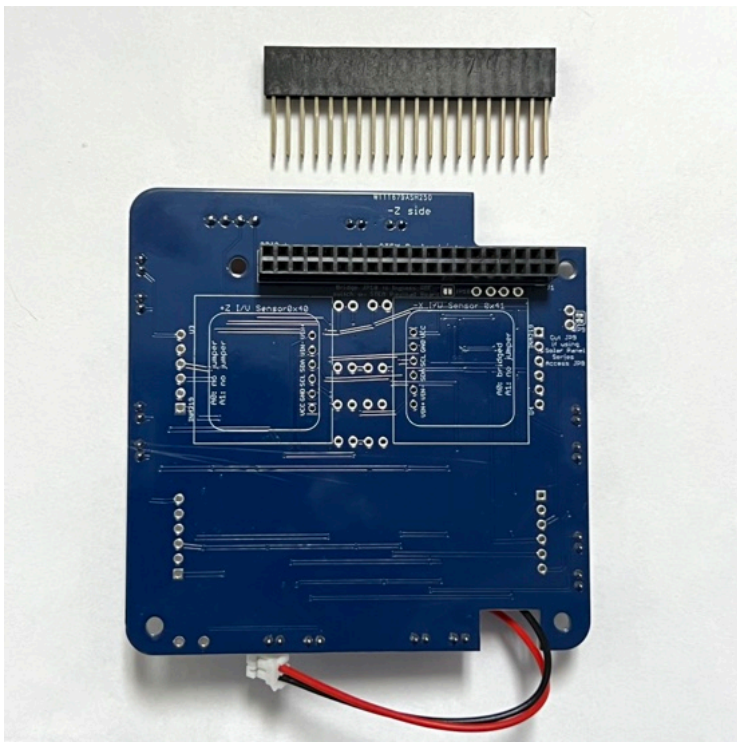
Here's how it looks from the sides:



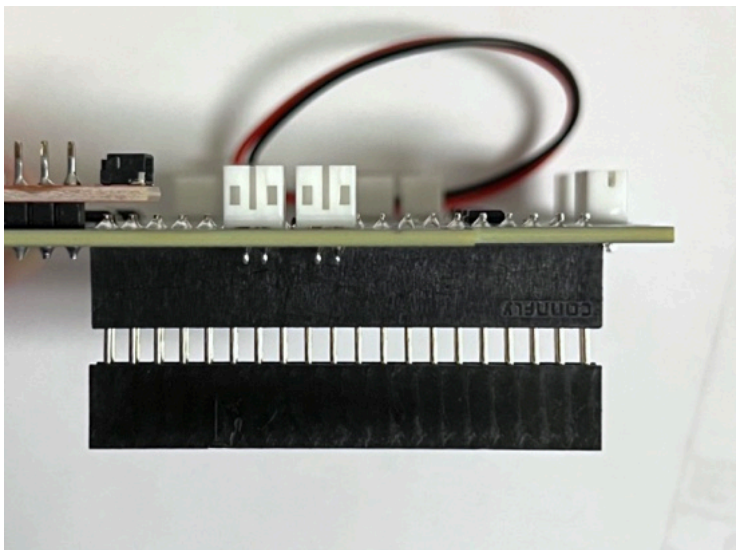


Next insert a stacking GPIO header into the bottom of the Solar Board:

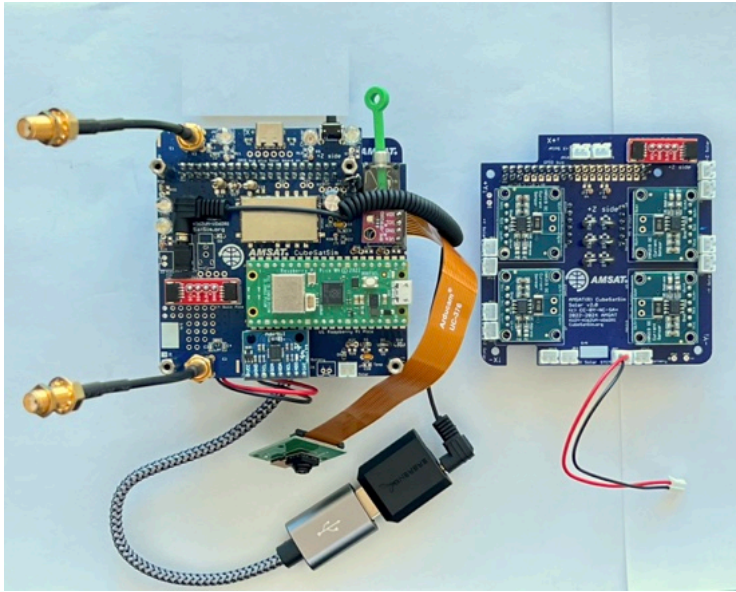




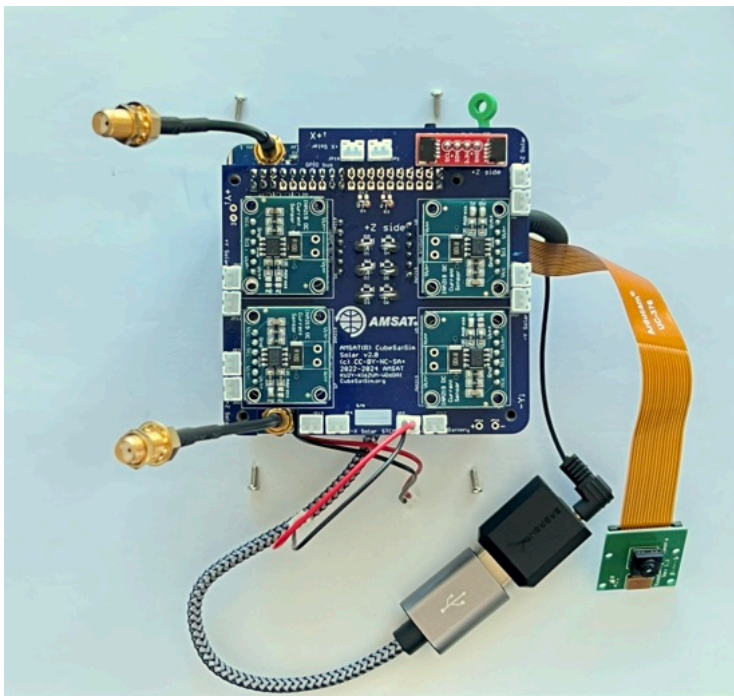




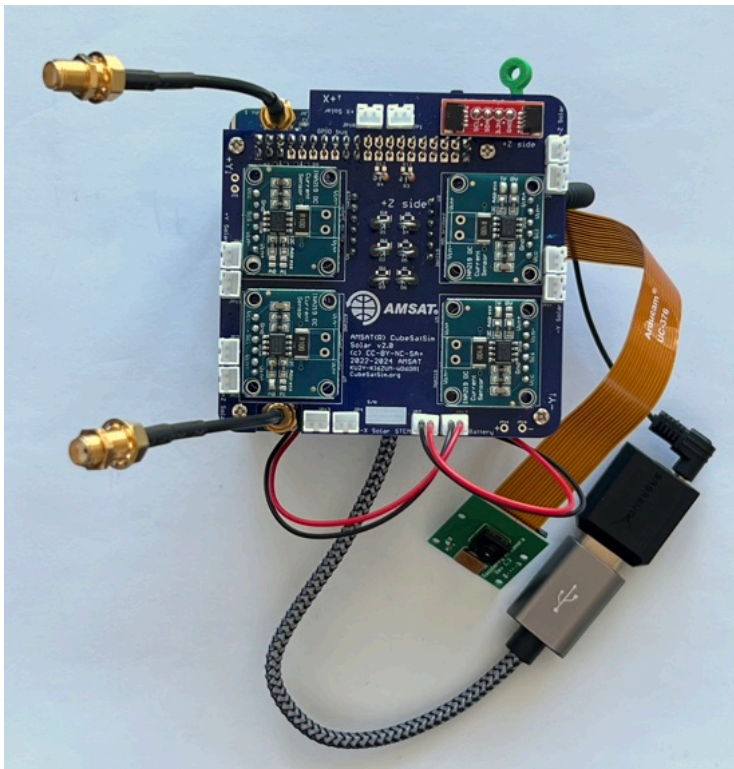
Plug the Solar board on top of the Main Board:



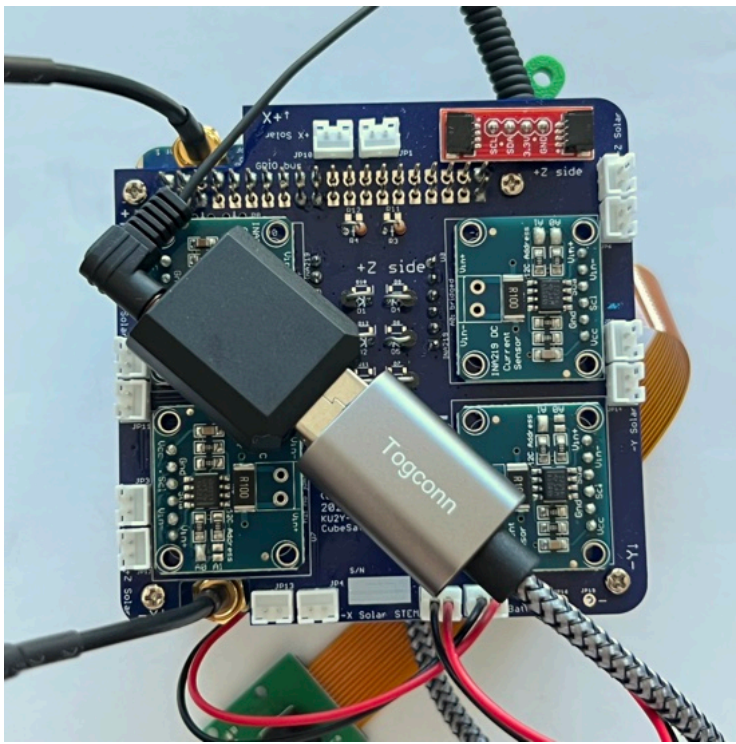
Secure with four M2.5 screws:



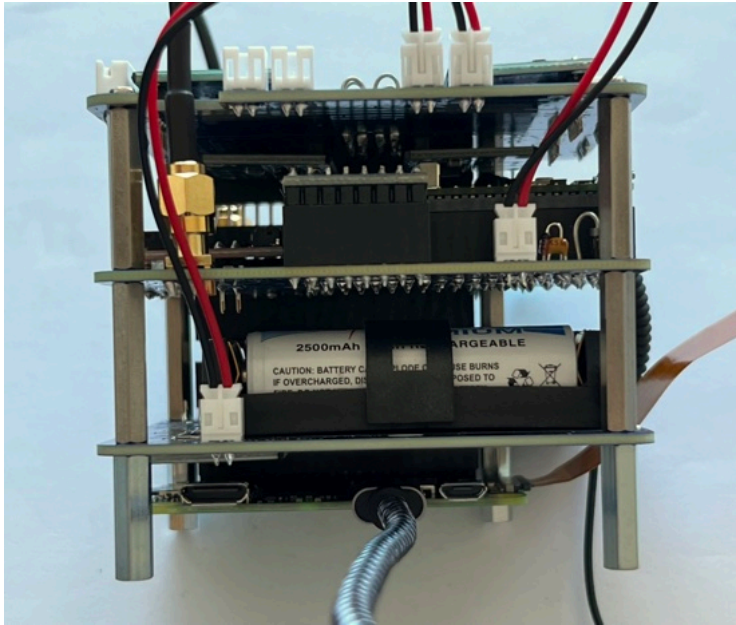
Connect the JST cable from the Battery board into the Solar Board. Connect the JST cable from the Solar board into the Main Board. It doesn't matter which connector on the Main Board you use.



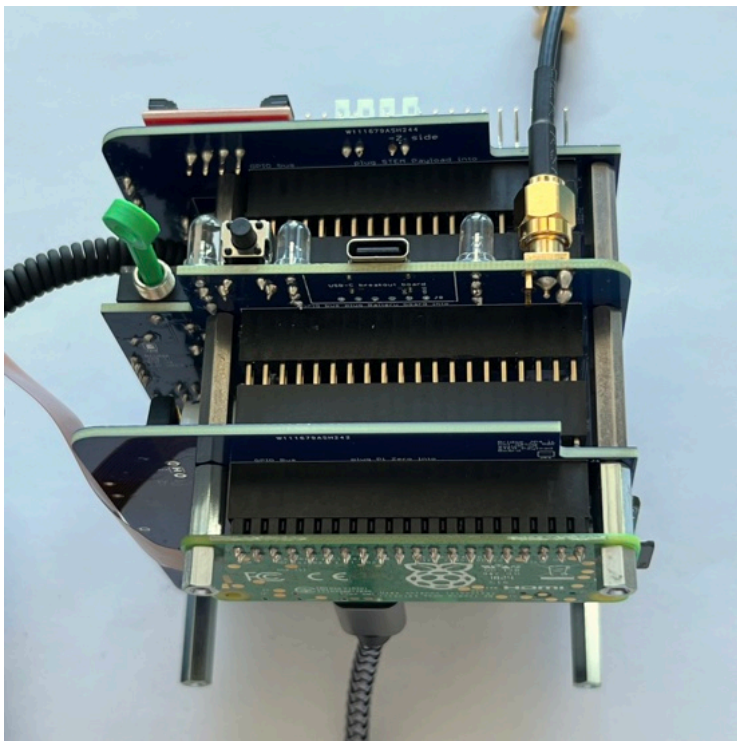
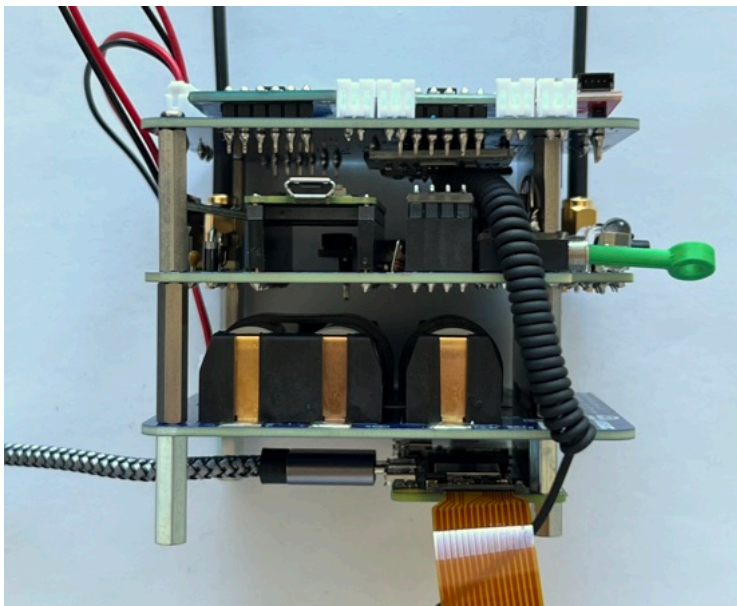
Here's how it looks with the sound card resting on top of the Solar board:



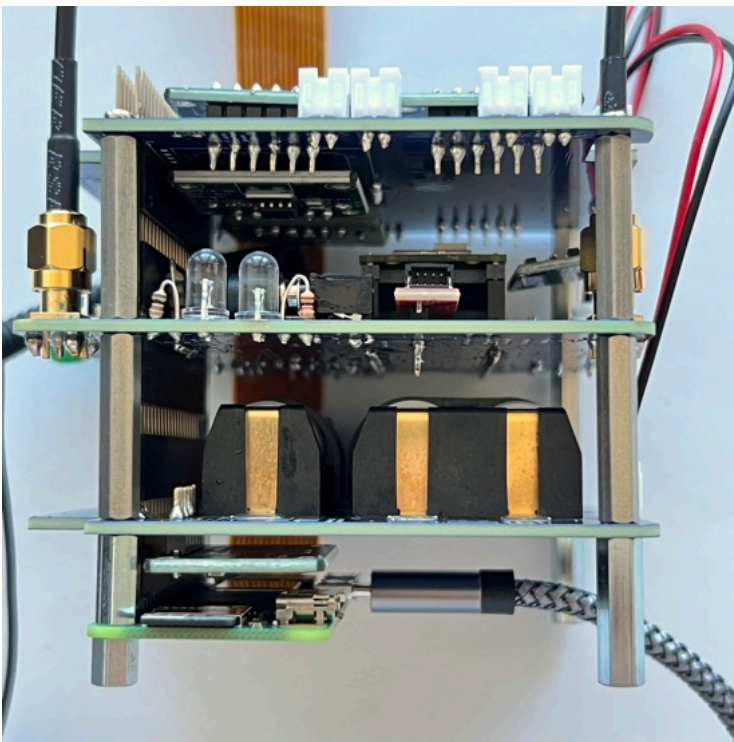
Here's how it looks from the side with the SMA antennas:











You can test the CubeSatSim by removing the RBF pin. Unless the batteries are fully depleted (NiMH batteries normally come charged up), the green light on the Pi will flicker and the CubeSatSim will boot up and start running.

The green LED on the Main Board will be lit up after booting is complete and the software is running. The blue LED on the Main Board will be lit up when the CubeSatSim is transmitting. The red LED on the Main Board will be lit up when the battery is being charged by either the micro USB power supply.

The pushbutton on the Main Board can be used to reboot or shutdown the CubeSatSim. Pressing and releasing the pushbutton will cause the green LED to turn off and the Pi to reboot. Pressing and holding the pushbutton until the green LED flashes three times slowly then releasing will shutdown the Pi. Once it has shutdown, it is safe to plug the RBF pin back in. Pressing the pushbutton with the RBF pin inserted will not turn on the Pi.

## 8.2 Installing the Board Stack into the Frame

---

The Board Stack is now ready to go into the Frame.

### Printing the CubeSatSim Frame

---

#### Frame Print

Here are the v2 Frame STL files:

<https://github.com/alanbjohnston/CubeSatSim/tree/master/hardware/frame/v2.0>

The frame has 4 parts. You need to print two of the top/bottom parts, and one of each side.

#### Frame Test Assembly with Solar Panels

---

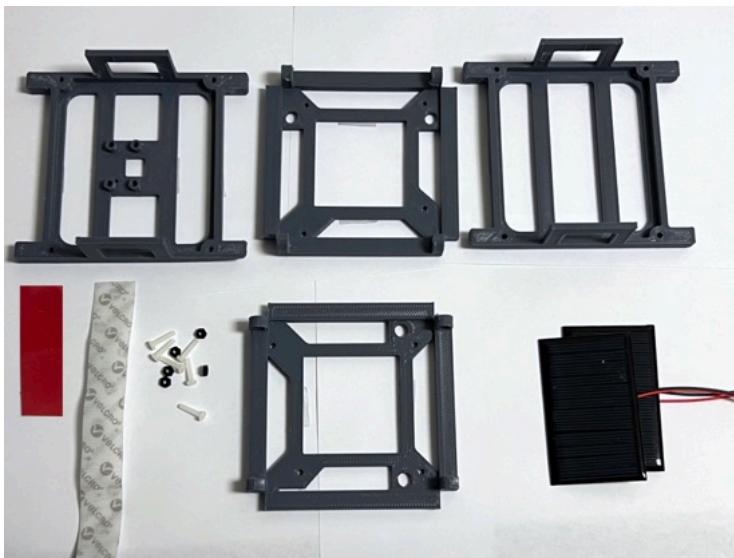
In this step, we will do a Frame test assembly and mount some of the Solar Panels on the Frame.

#### Video

---

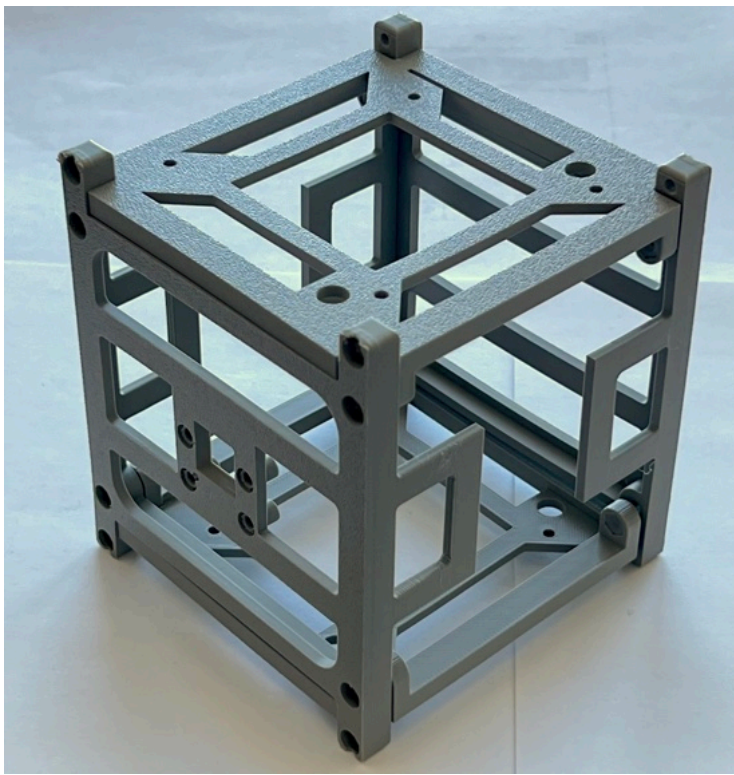
A video of [this step is available here.](#)

Here's the frame parts with the ten Solar panels, plastic screws and nuts, and tape/velcro to secure the panels to the frame:

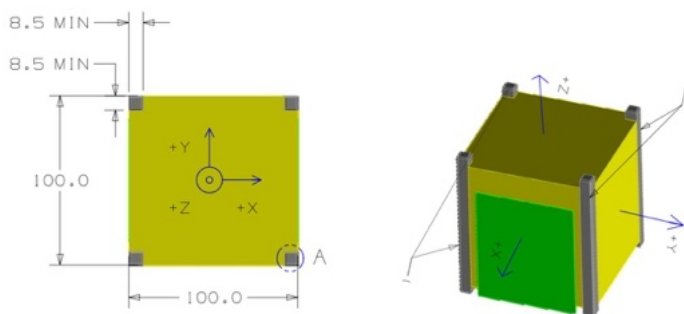


The red is double stick tape which is used to attach the solar panels more or less permanently. You can remove the panels after sticking them, but it isn't easy and can only be redone once or twice. I'd recommend using the red double stick tape for the panels that never need to be removed for disassembly. For the four panels that need to be removed to disassemble the frame, I'd recommend using the white velcro (hook and loop tape) instead.

Here's the frame when it is put together:



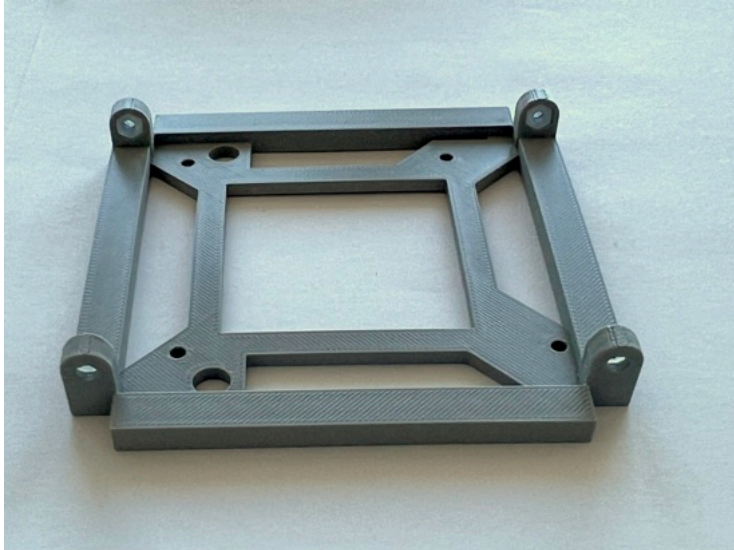
The 1U frame of the CubeSatSim has solar panels on each face, which are labeled by the Cartesian axes X, Y, and Z. For the sides that face in the direction of the axes (see [https://en.wikipedia.org/wiki/Cartesian\\_coordinate\\_system#/media/File:Coord\\_system\\_CA\\_0.svg](https://en.wikipedia.org/wiki/Cartesian_coordinate_system#/media/File:Coord_system_CA_0.svg)), they are labeled +X, +Y, and +Z. For the sides that face in the opposite direction, they are labeled -X, -Y, and -Z. The CubeSat Design Specification (CDS) standard also has drawings showing the X, Y, and Z sides of a 1U CubeSat <https://www.cubesat.org/cds-announcement>. Here is a figure from that document:





On the CubeSatSim, the **+Z side** is the top of the CubeSatSim, the **+X side** has the pushbutton, micro USB connector, and the LEDs. The Main printed circuit board has the axes labeled on it as well.

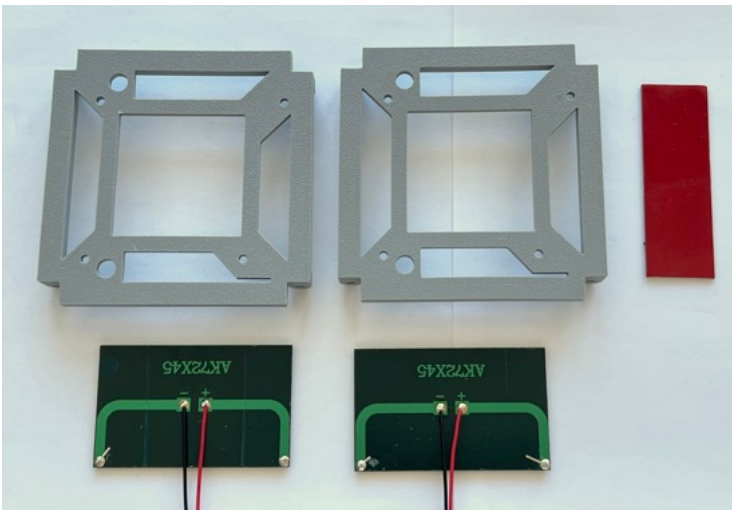
Start by pressing the eight M3 nuts into the top and bottom frame parts.



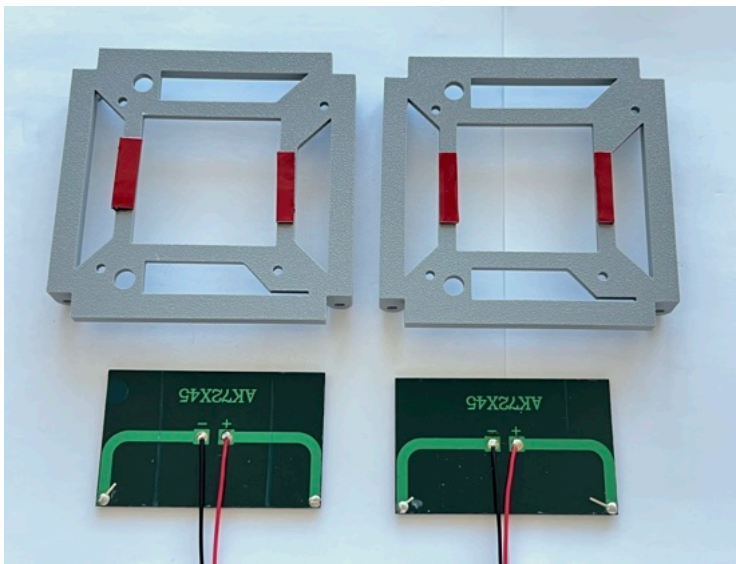
Sometimes, it takes more than finger pressure - use needle nose pliers to push them in:



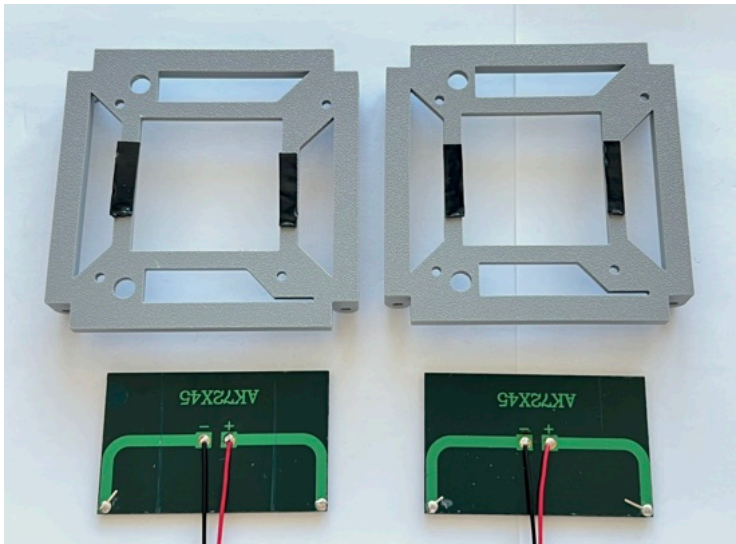
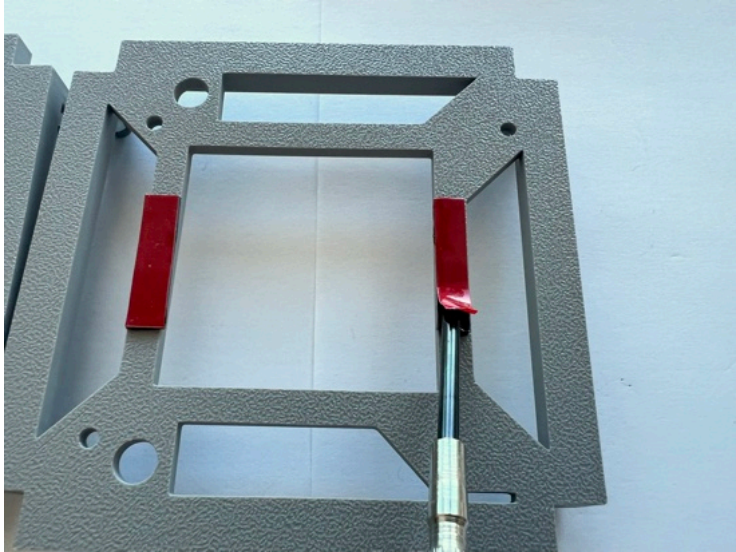
We will start mounting the solar panels with the top and bottom of the frame (the +Z and -Z sides). One solar panel is mounted on each.



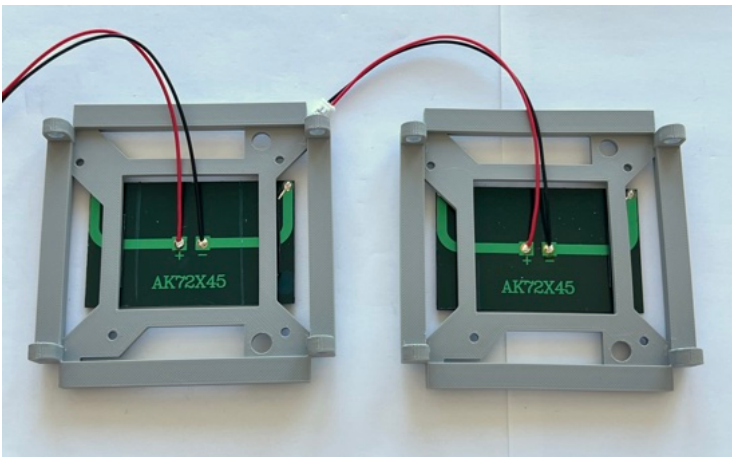
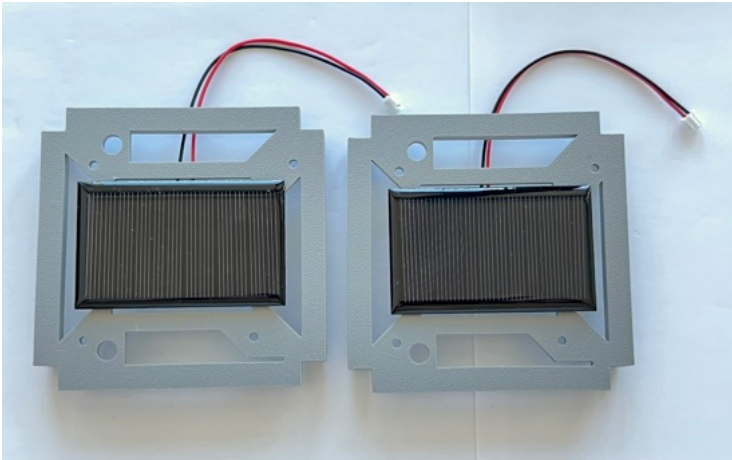
Cut four strips of double stick tape. Peel the backing on one side and stick them on the ribs on the frame:



Peel the backing on the other side. Sometimes a flat screwdriver or a blade can be helpful:



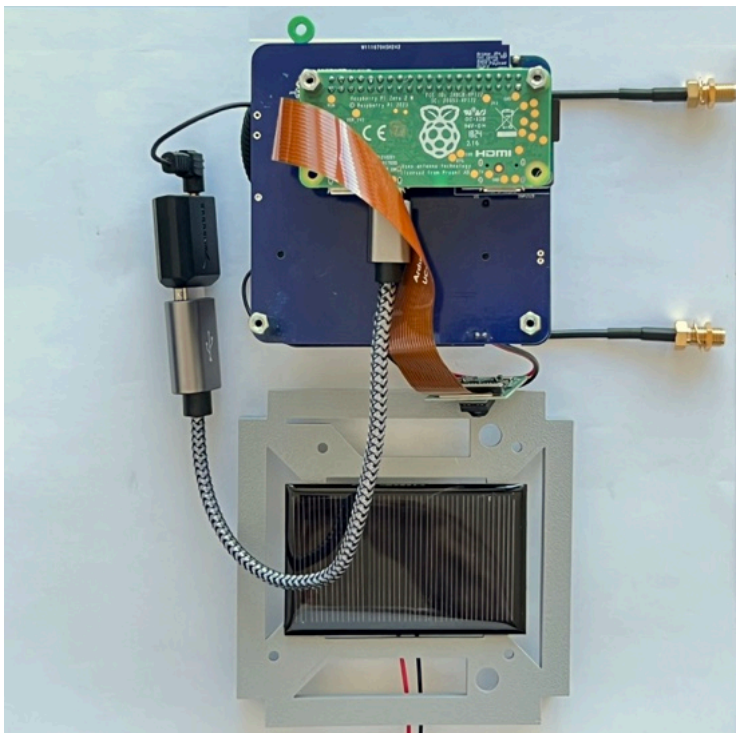
Stick the solar panels onto the frame, making sure the JST connectors go to the inside of the frame. Here's how they look when mounted:



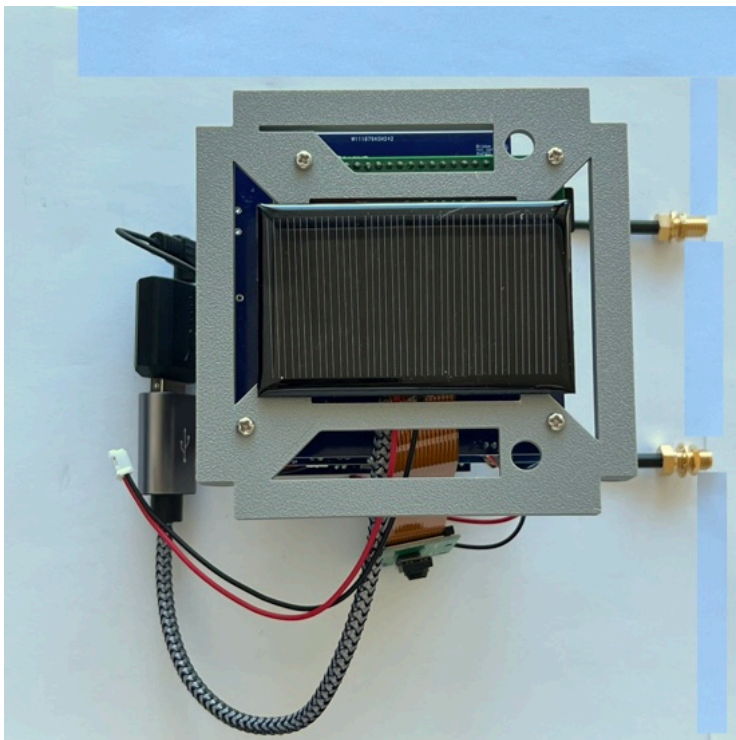
We will now attach the board stack to the bottom frame, then attach the two sides, then the top.

Flip the board stack upside down so the Pi Zero 2 is on top. Put the -Z frame bottom next to it as shown here. Make sure the LEDs and other connectors are facing to the top as in this photo. The Pi Camera ribbon cable is routed to the bottom as shown here:

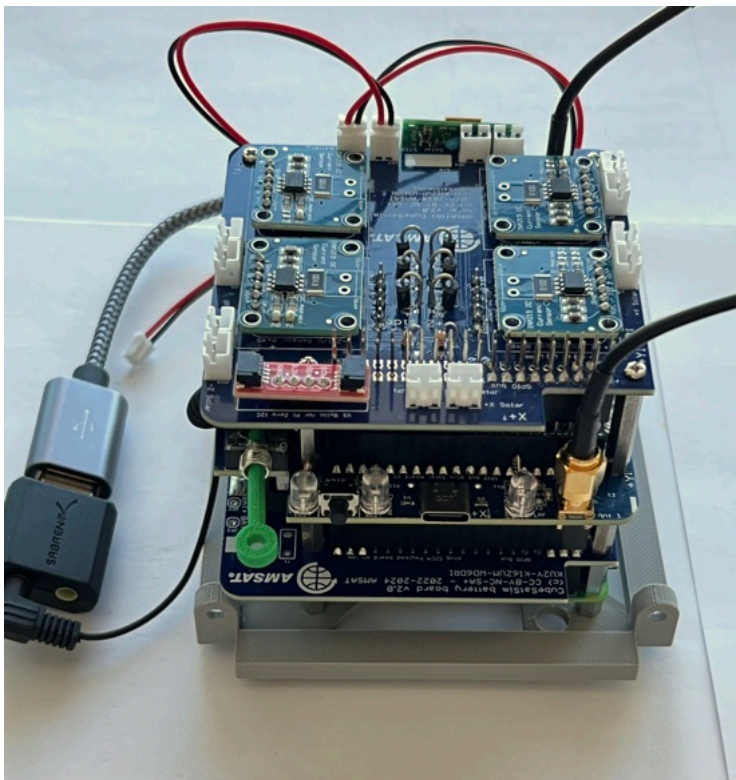




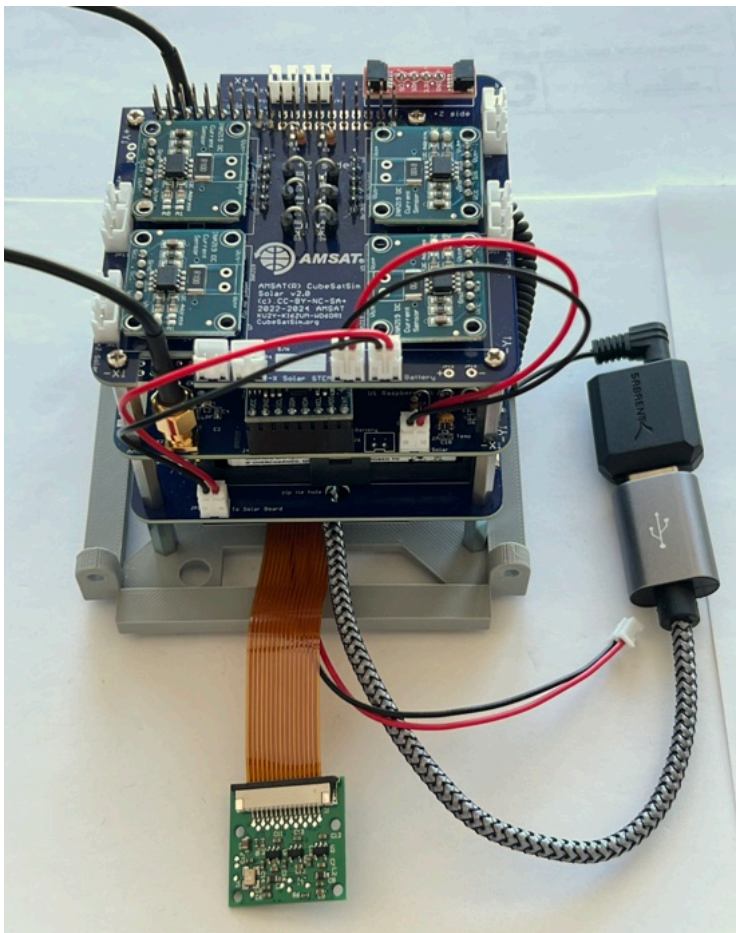
Place the -Z frame on top and secure with four screws:



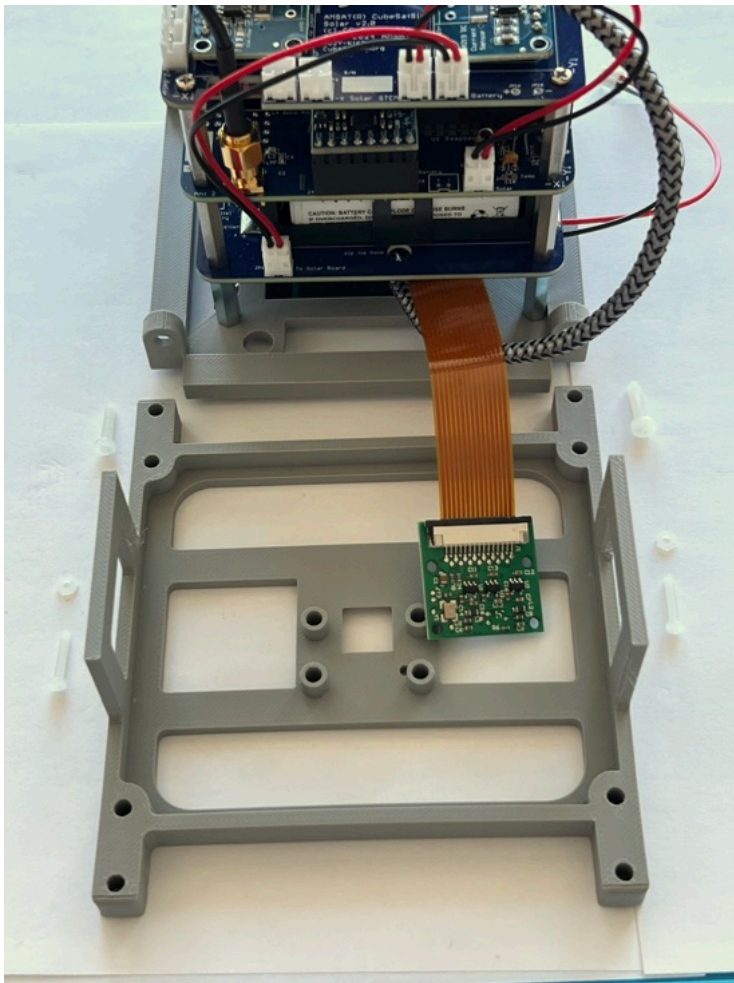
Flip it so the -Z frame is on the bottom. Here's how it looks from the +X side, the side with the LEDs and connectors.



The Pi Camera should face outward on the -X side opposite the LEDs and switches as shown here:



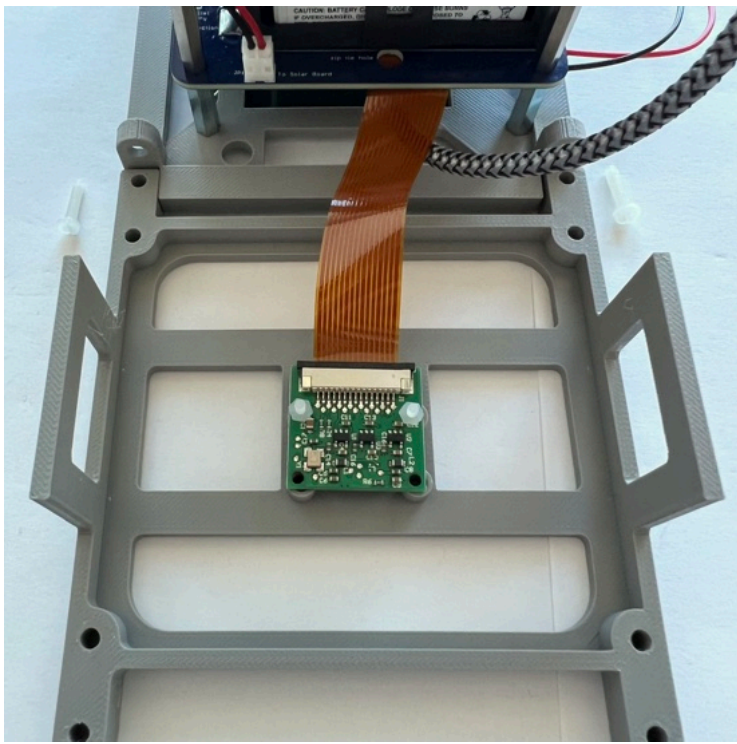
Place the frame side with the camera mount next to the **-X side** as shown - make sure the frame isn't upside down:



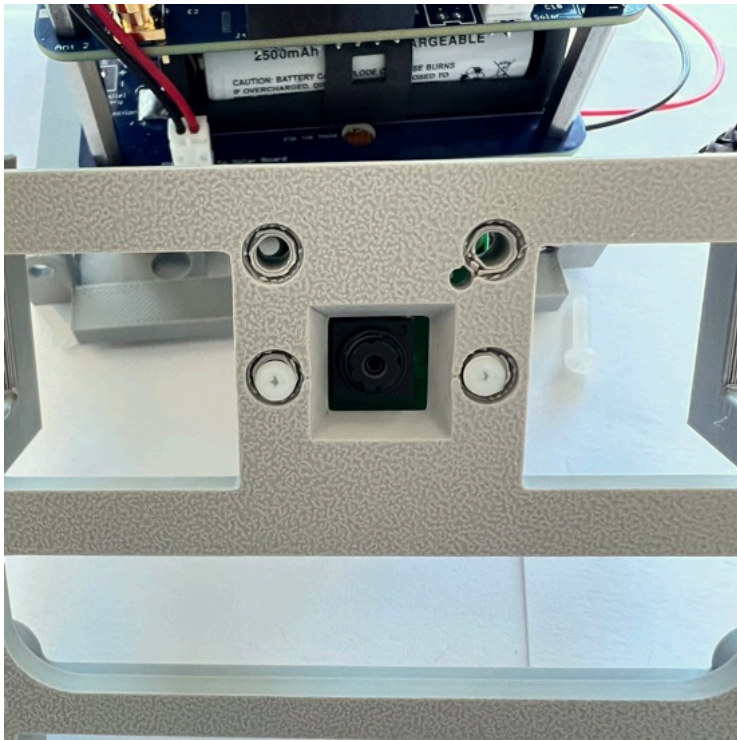
If you are using the SMA antennas, make sure the SMA coax cables are screwed into the Main board.

The first frame side to be mounted is the **-X side** which has the Pi Camera mount. We will mount the Camera using two M2 nylon screws and two M2 nylon nuts (Note: you can use four screws and nuts, but you do not need the other two and you won't be able to unscrew the camera without removing a solar panel in the future):



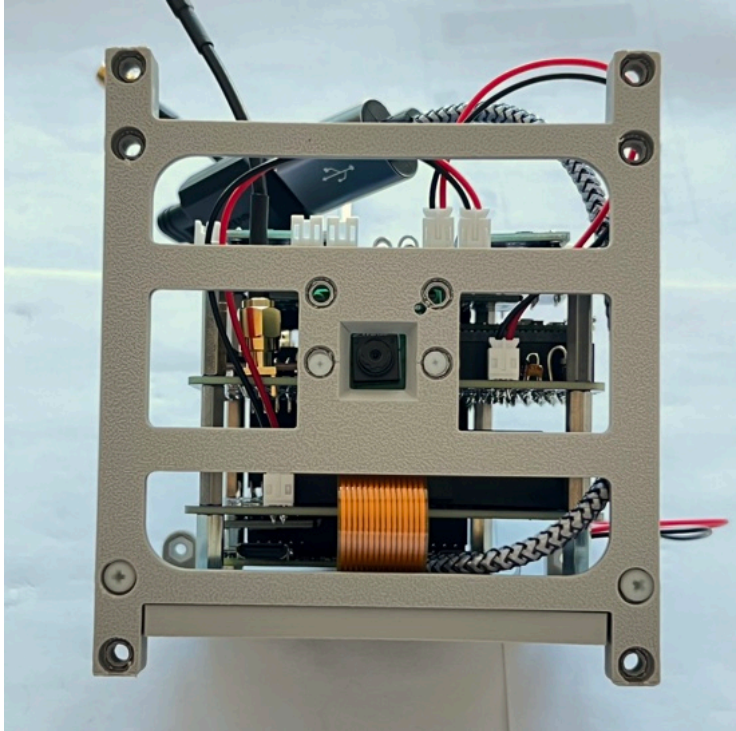


Here's how the Camera looks from inside and outside the frame when it is mounted (Note: again, although four screws are shown, you only need the bottom two screws):

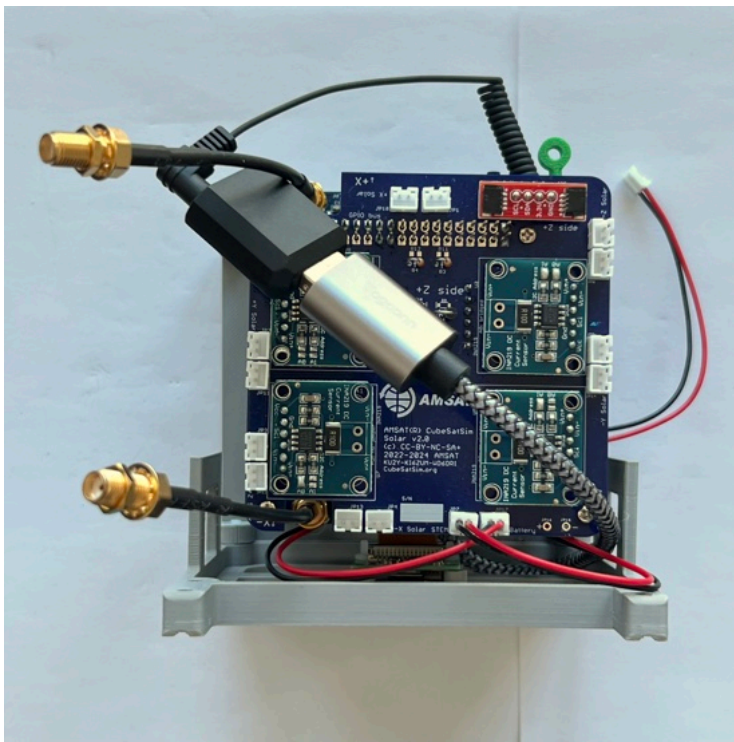




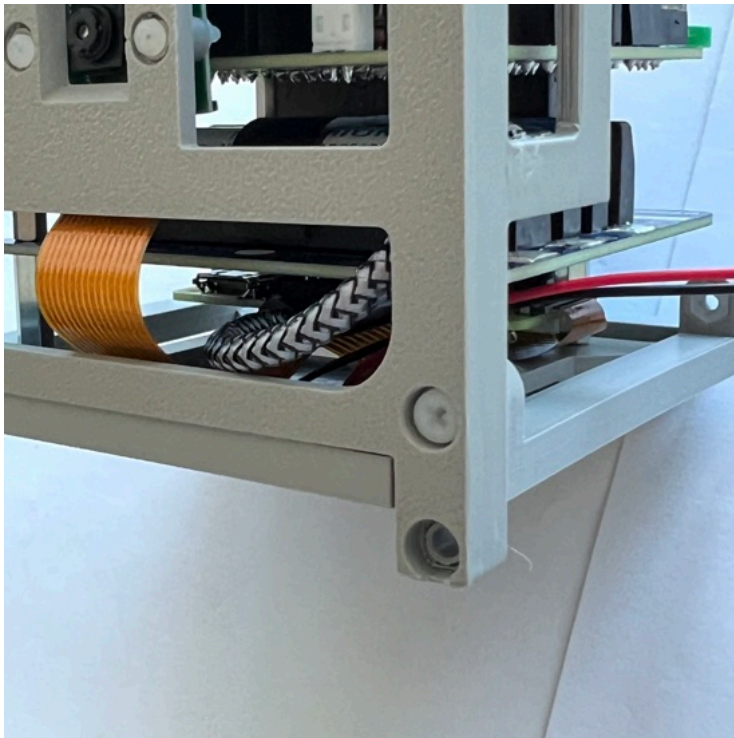
Carefully stand up the frame, being careful not to stress the Pi Camera ribbon cable and snap into position on the bottom frame. Secure the -X frame with two nylon screws and two nuts to the -Z bottom frame. Here's how it looks from the outside:



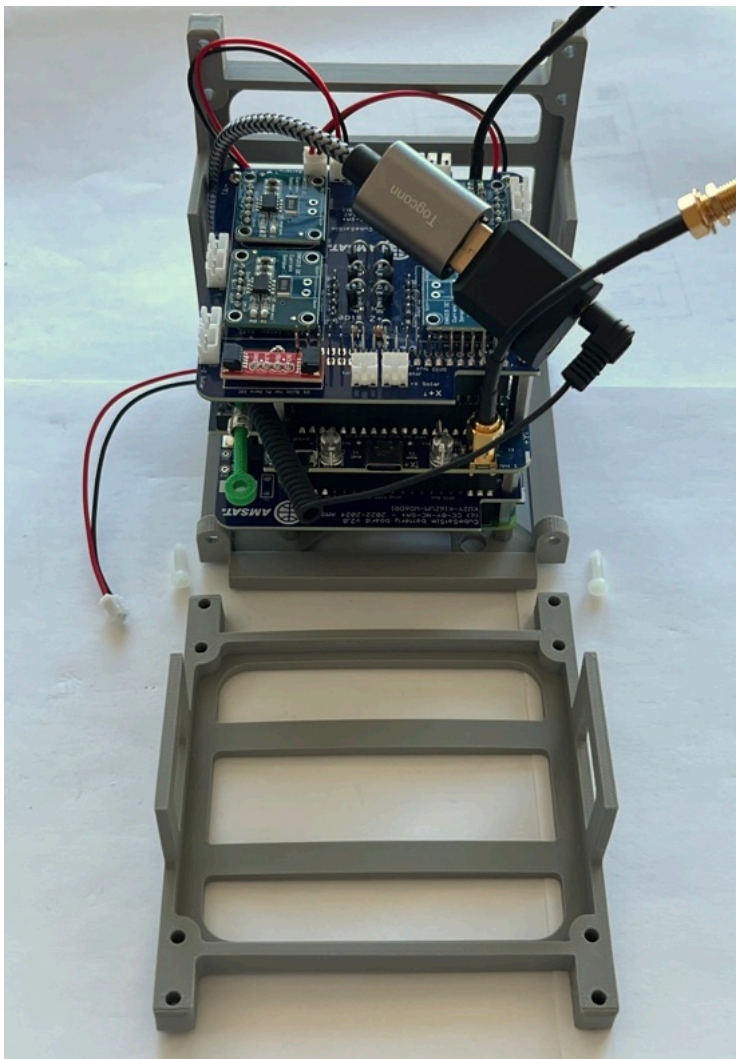
Make sure the USB OTG cable is routed inside the frame as shown. Here's how it looks from the top:



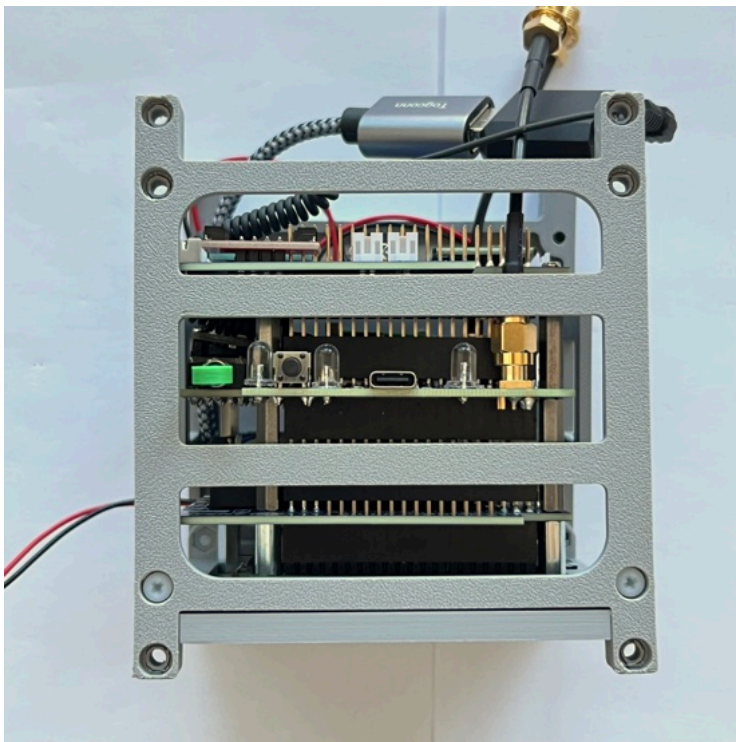
Here's how the camera ribbon and USB OTG cable should be routed:



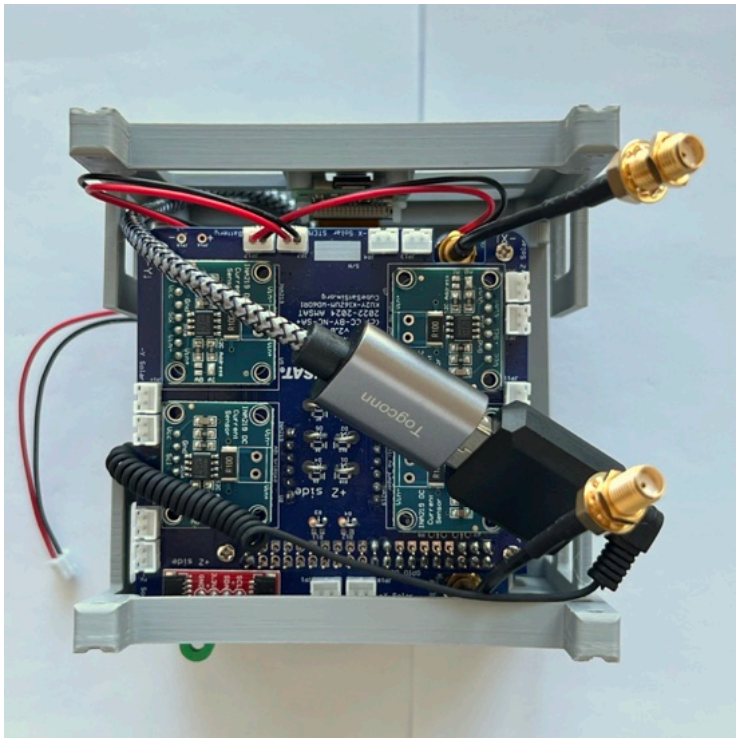
The next frame side to be mounted is the **+X side**. We will use the frame without the camera mount and two nylon screws.



Snap the side frame onto the bottom frame and secure the side frame to the bottom frame with the two screws.

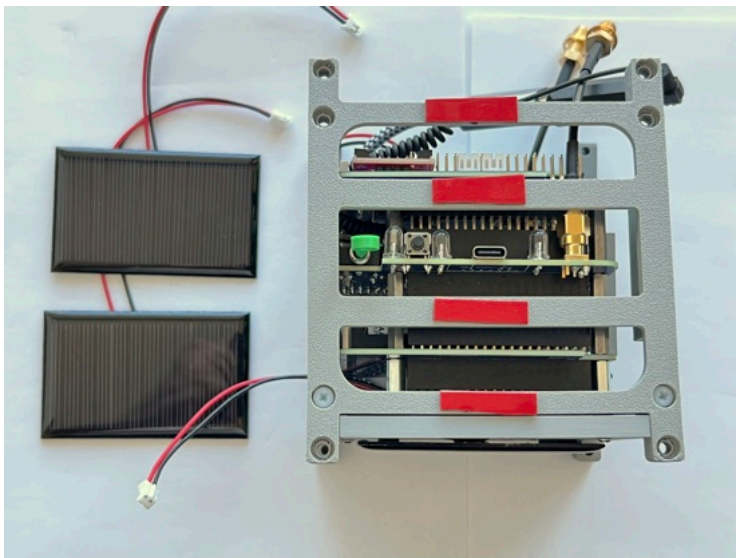


Here's how it looks from the top:

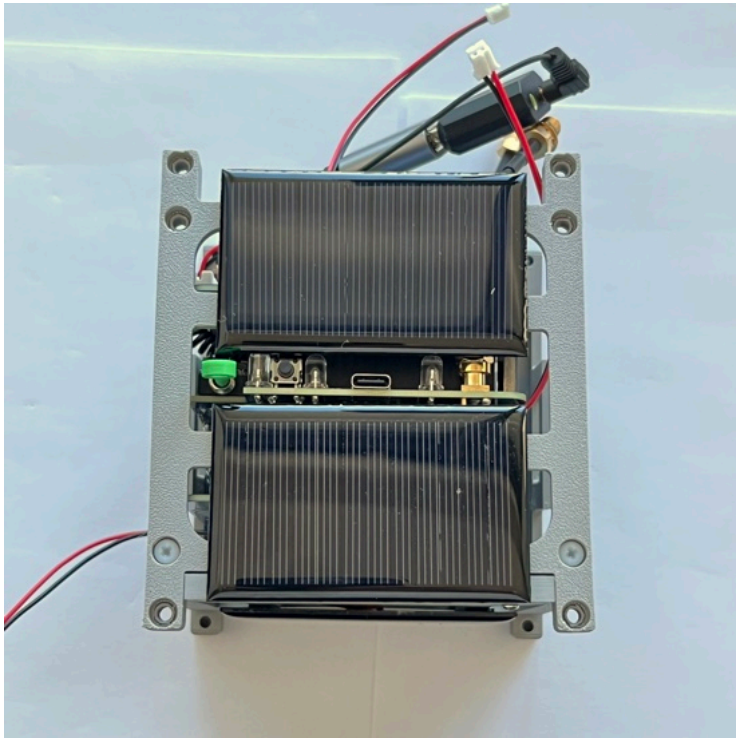


Next, secure two solar panels to the frame using 4 pieces of double stick tape:

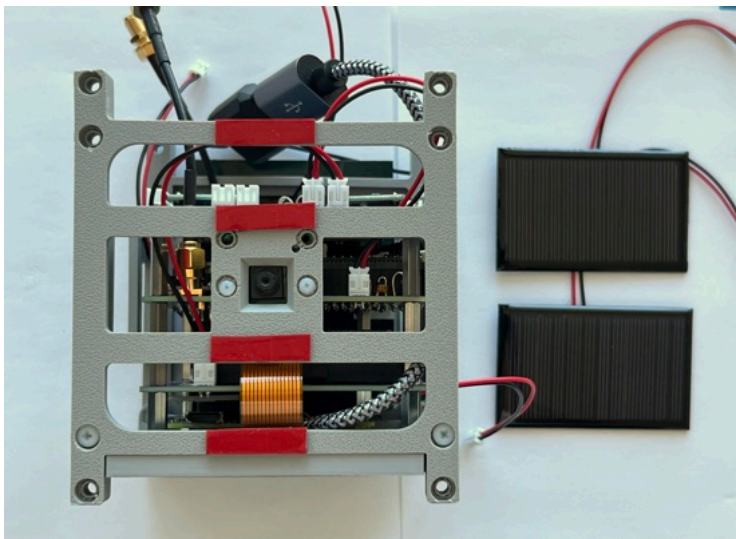




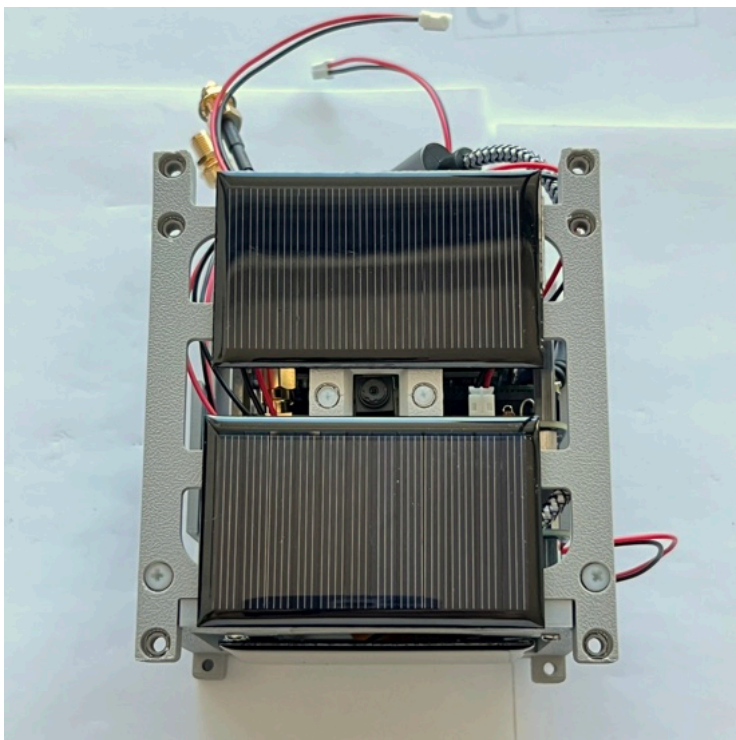
Make sure there is room to see the LED and to access the USB-C and push button:



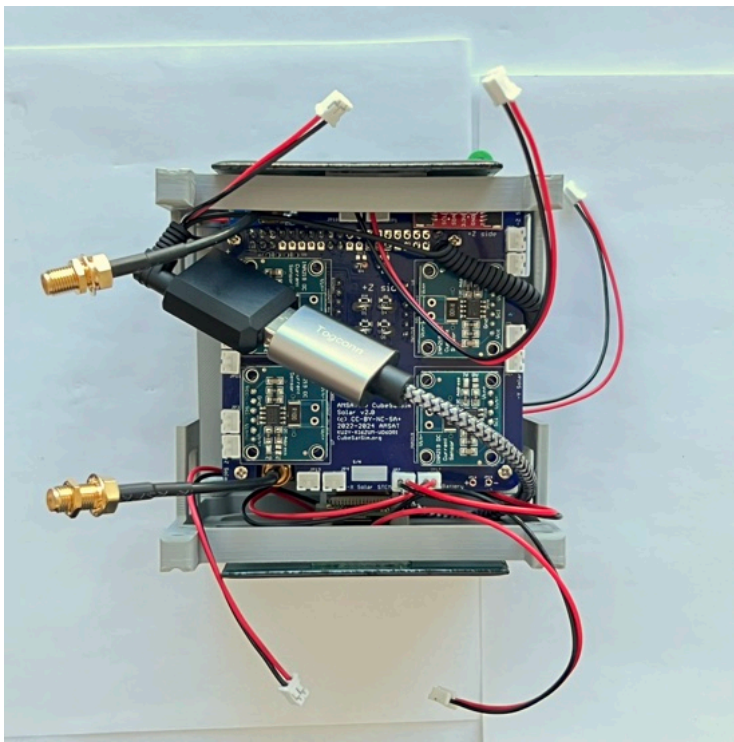
Next, do the same on the -X side frame:



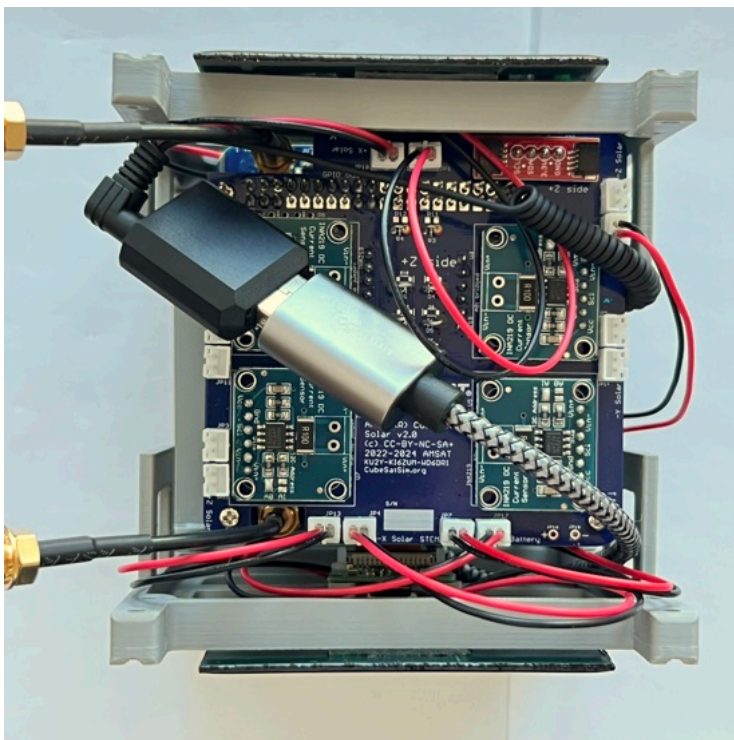
Mount the solar panels close to the camera as shown:



Here's how it looks from the top:

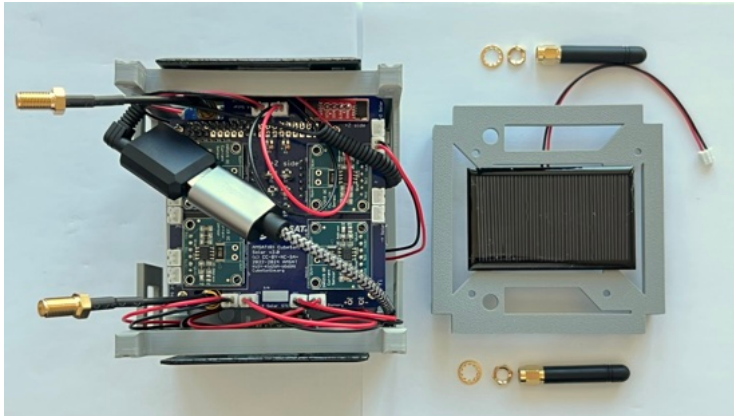


Plug in the two +X JST cables and the two -X JST cables and the -Z JST cable into the Solar board. Here's how it looks from the top with both sides attached and the solar panels plugged in:

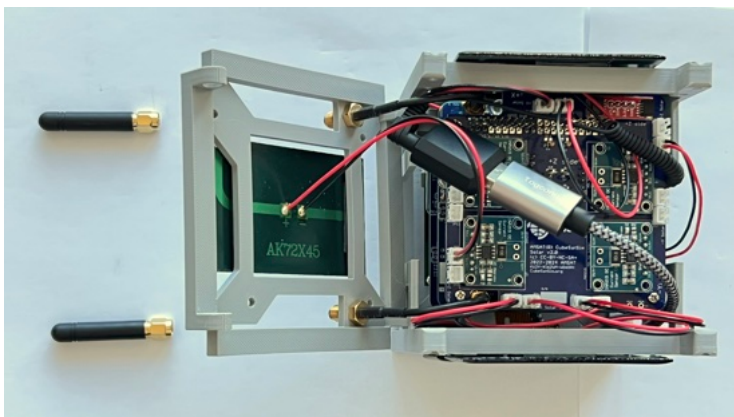
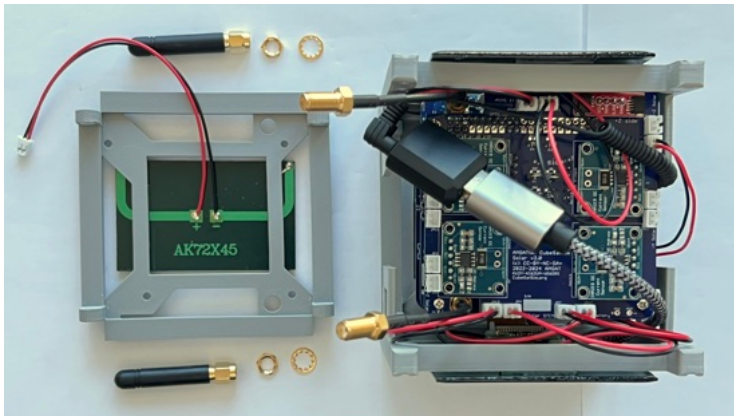




Next, we will connect the top frame +Z, but we won't secure it with screws until all the solar panels have been plugged in:

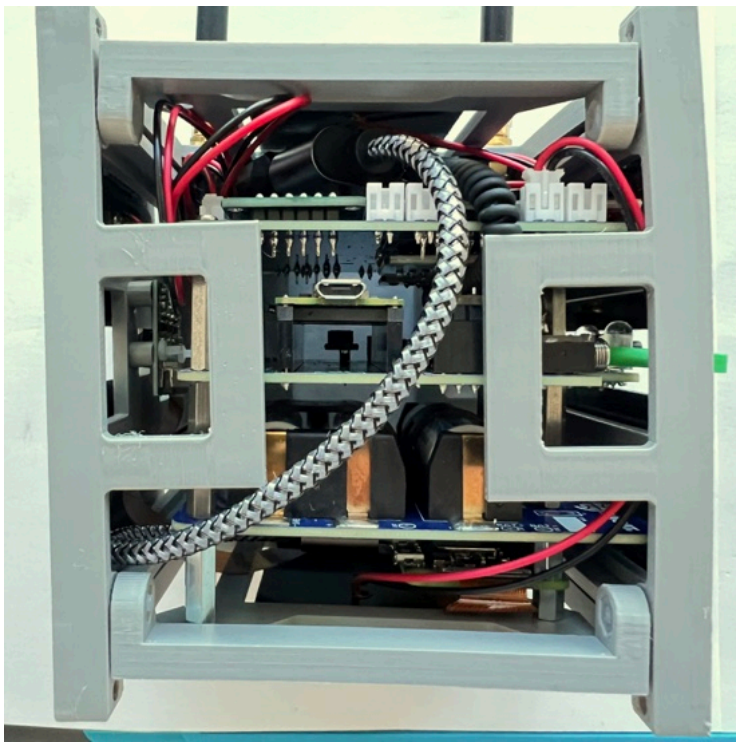


If you are using the SMA antennas, remove the nuts and washers from the coax and insert the coax through the frame, then put the nut on the other side to secure the coax. Plug in the +Z JST cable into the Solar board:



Then, carefully bend the coax so that you can put the top frame in place:



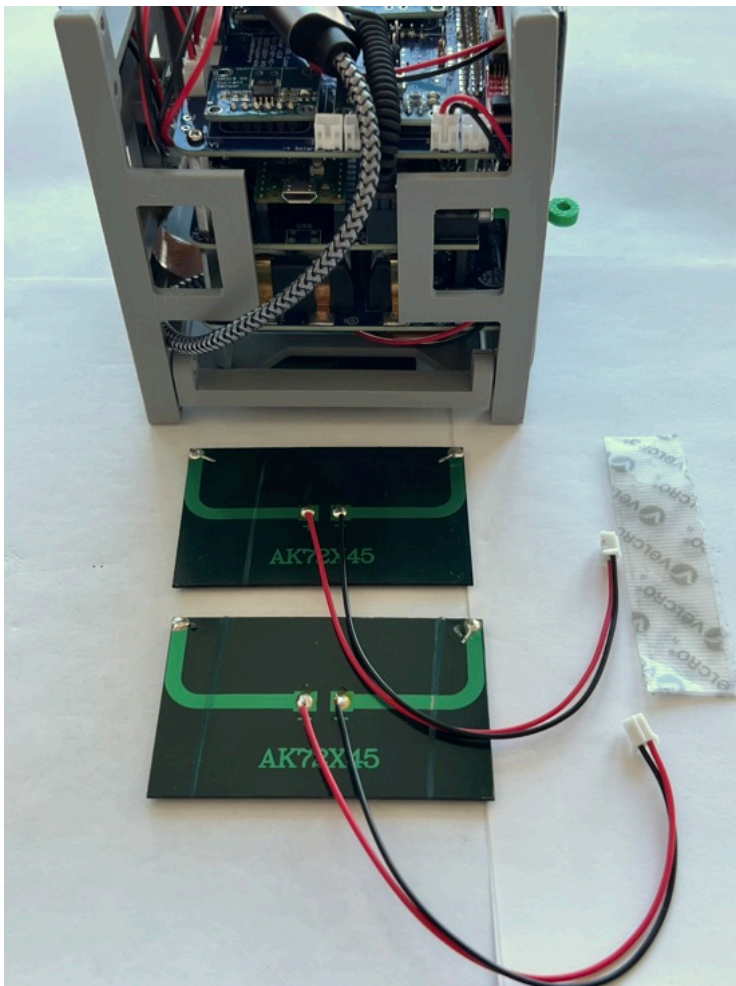


Don't secure the top frame to the sides until all the solar panel connections have been made.

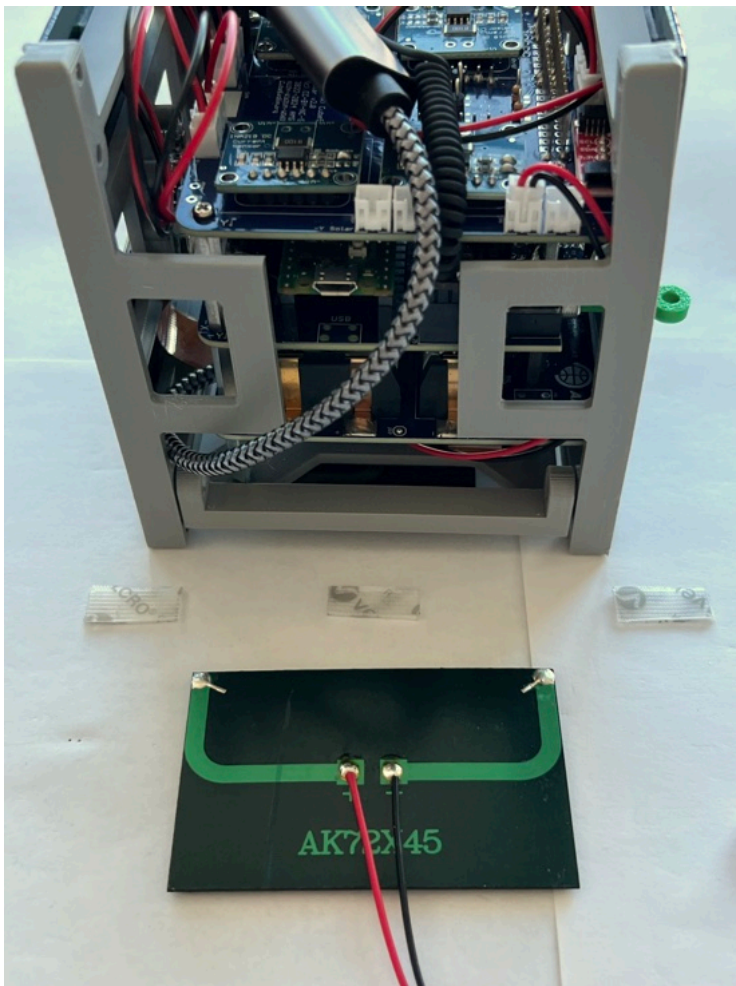
This video shows how the USB sound card should be mounted and the OTG cable routed so that it isn't in the way of the solar panels:

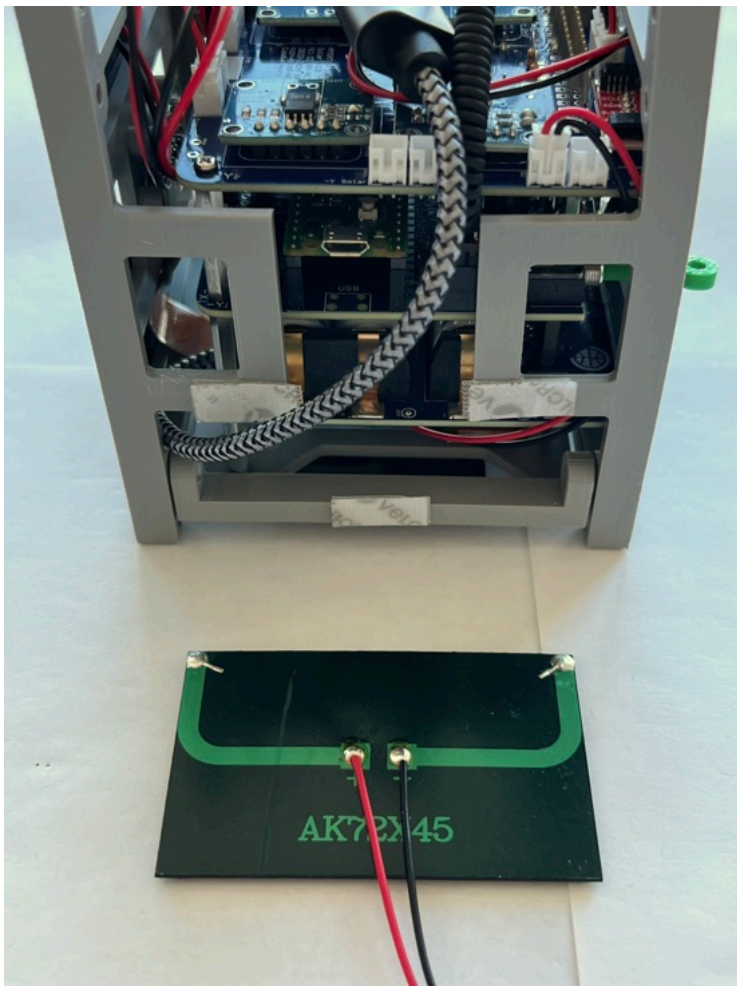
[https://cubesatsim.org/v2/stacking/Sound\\_Card\\_USB\\_Routing.MOV](https://cubesatsim.org/v2/stacking/Sound_Card_USB_Routing.MOV)

Now we will attach the -Y solar panels with pieces of velcro (you can use double stick tape, but these panels will need to be removed in order to disassemble the frame in the future):

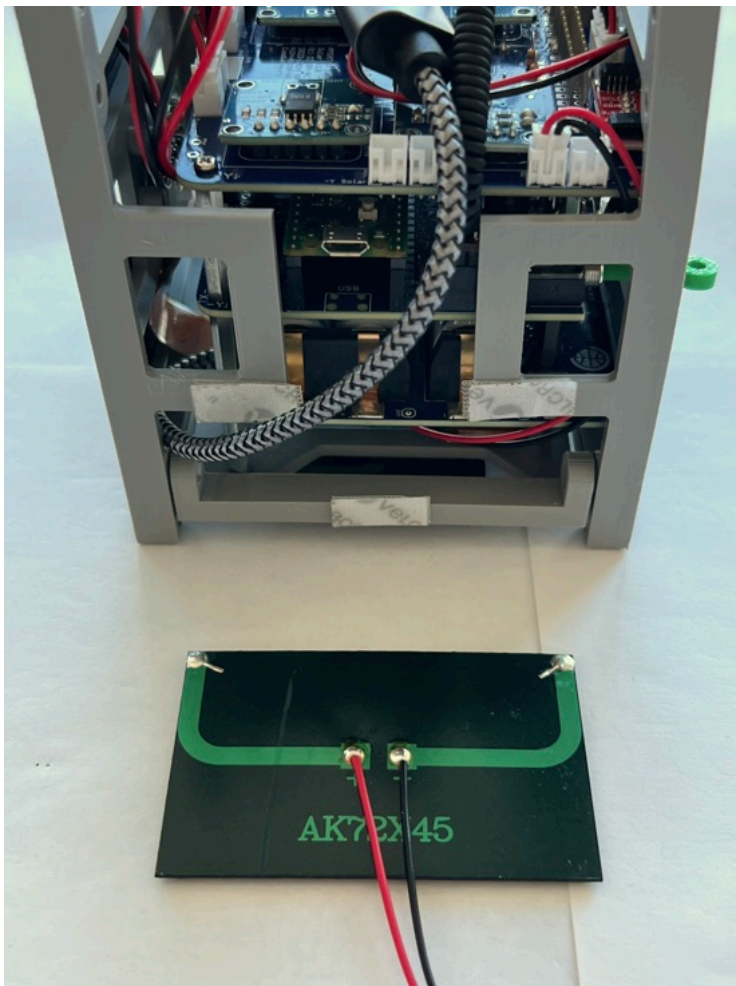


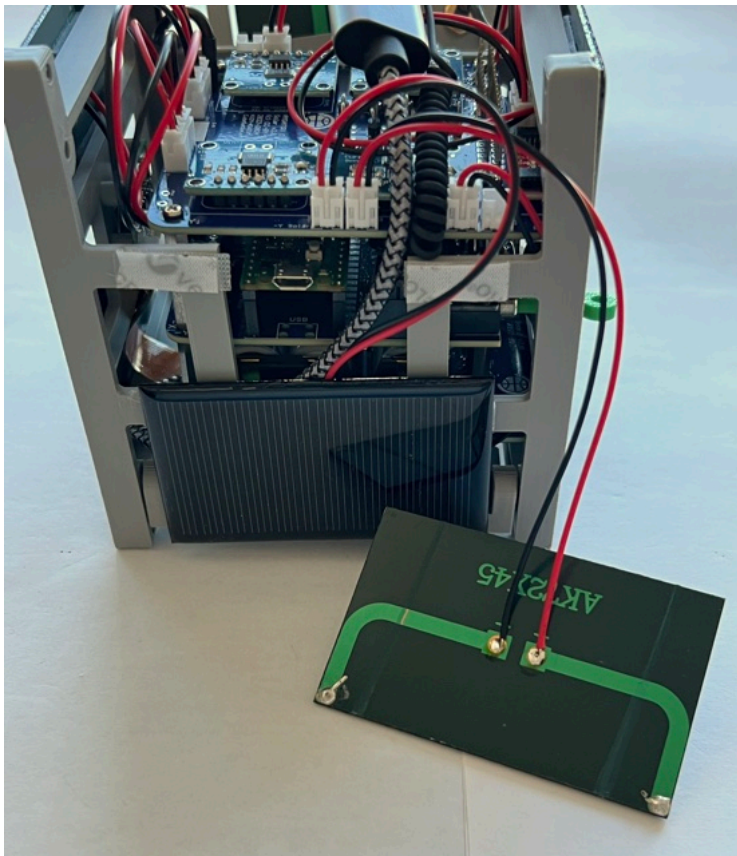
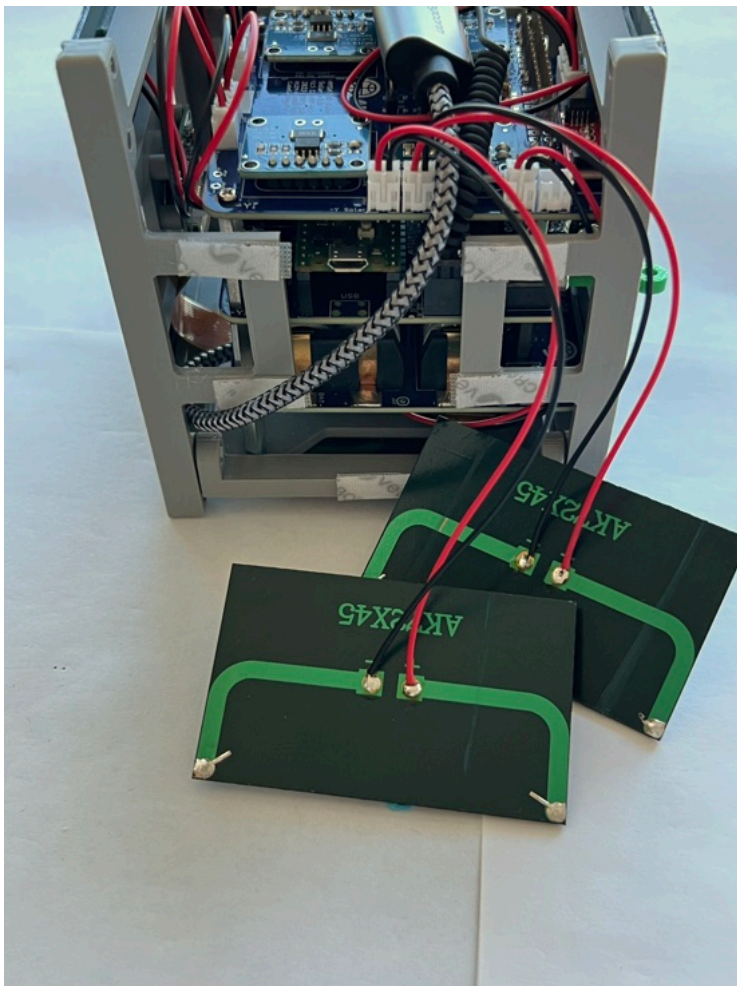
Stick the tape/velcro to the frame as shown:

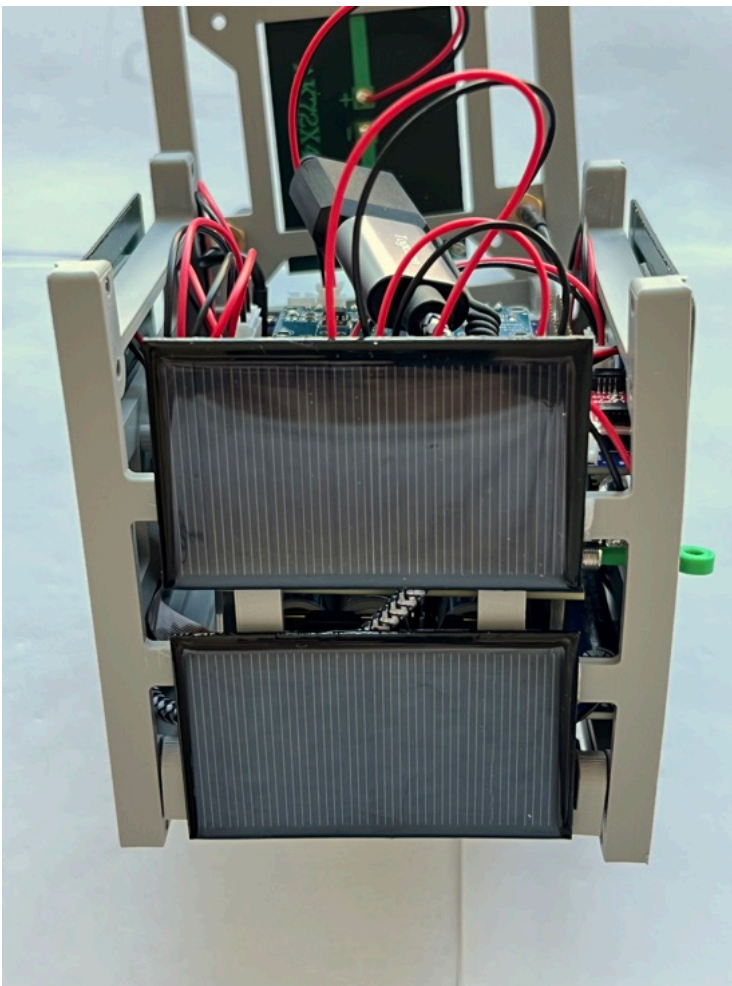






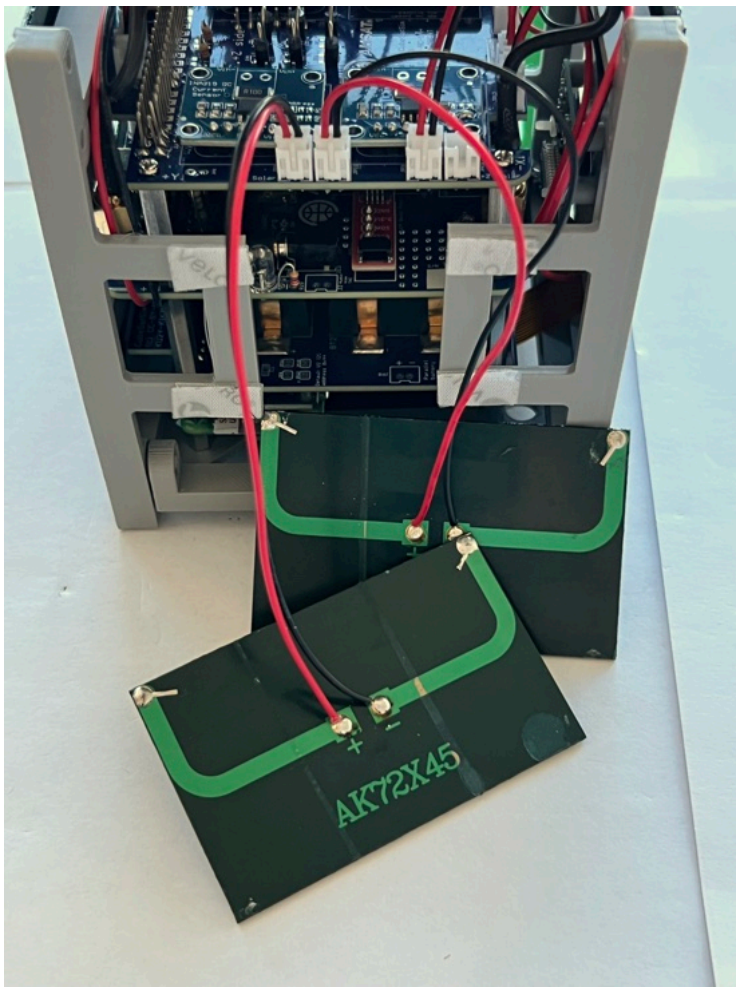




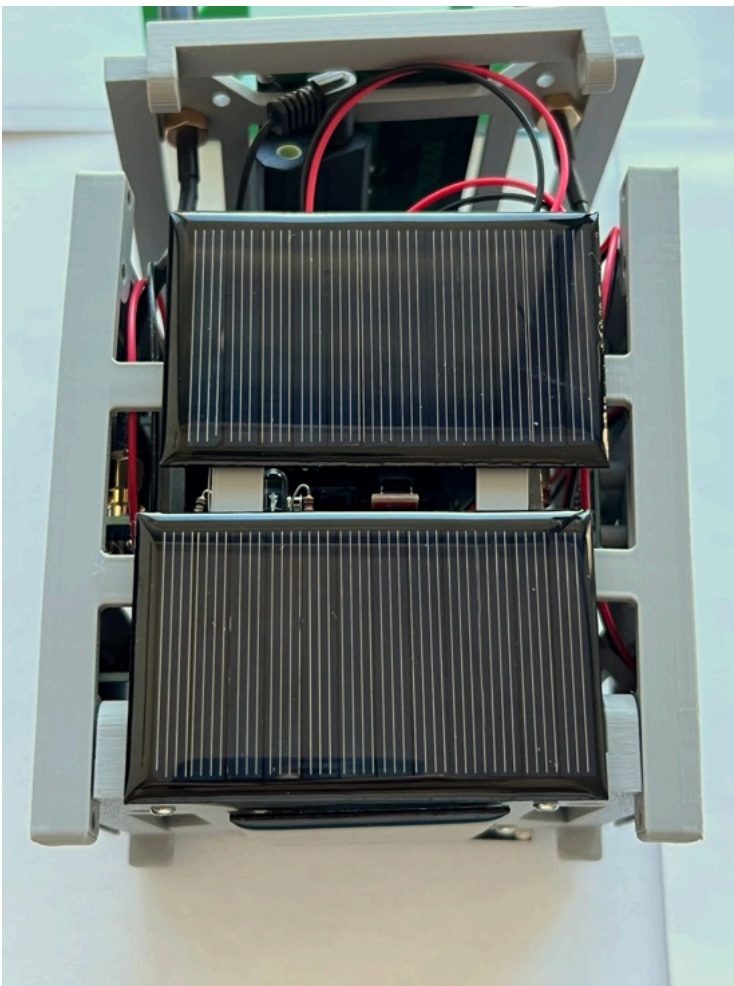


Plug the two -Y JST solar panel connections to the Solar board.

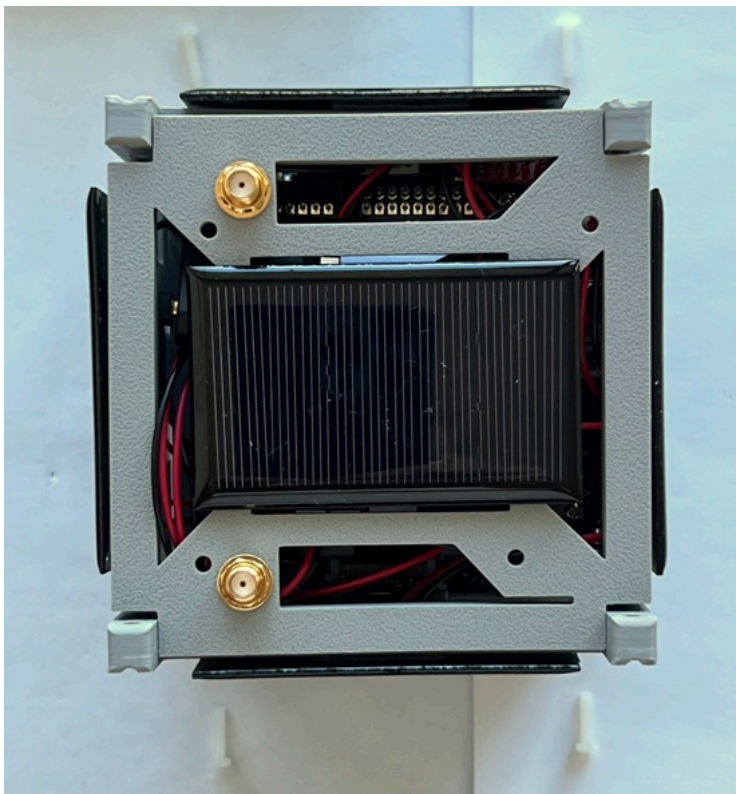
Now do the same thing for the +Y solar panels. First stick the tape/velcro to the frame. Then connect the +Y JST connectors to the Solar board. Then stick the panels onto the frame.



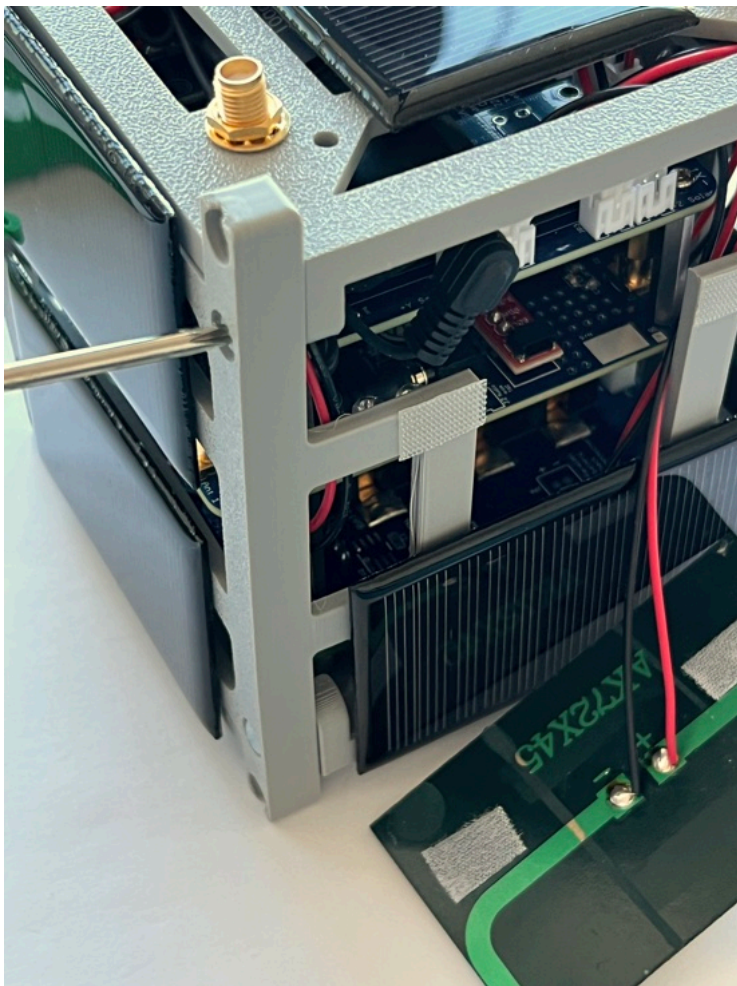




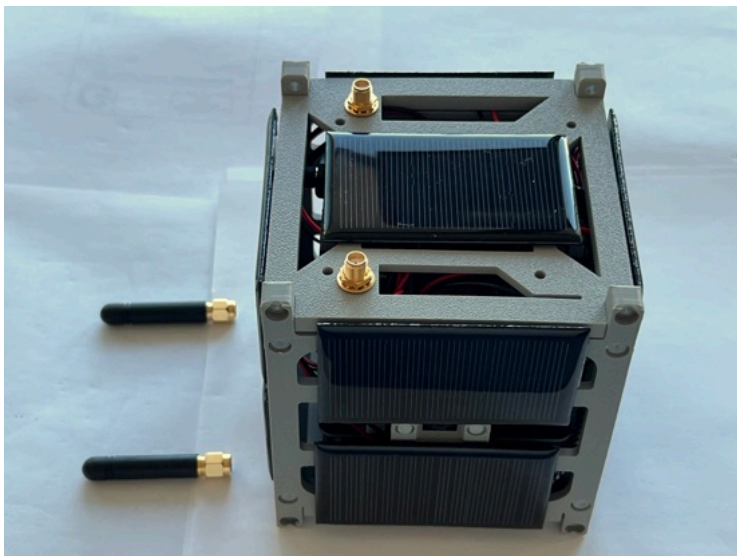
Now you can secure the top frame using four plastic screws and four nuts

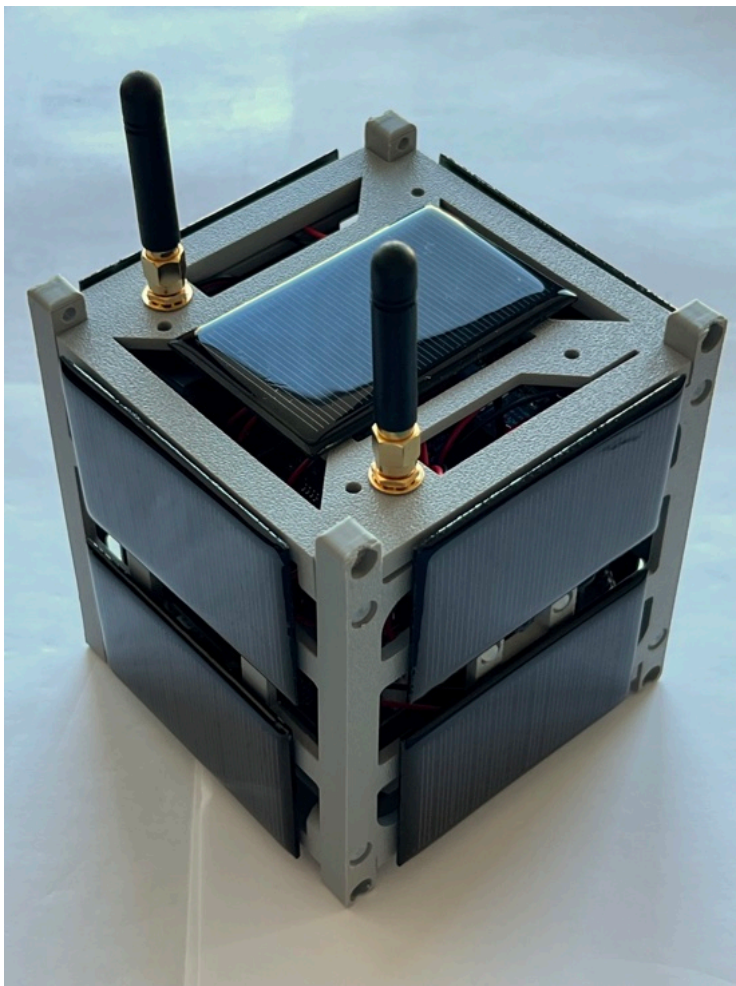


You might need to remove a solar panel to do this:



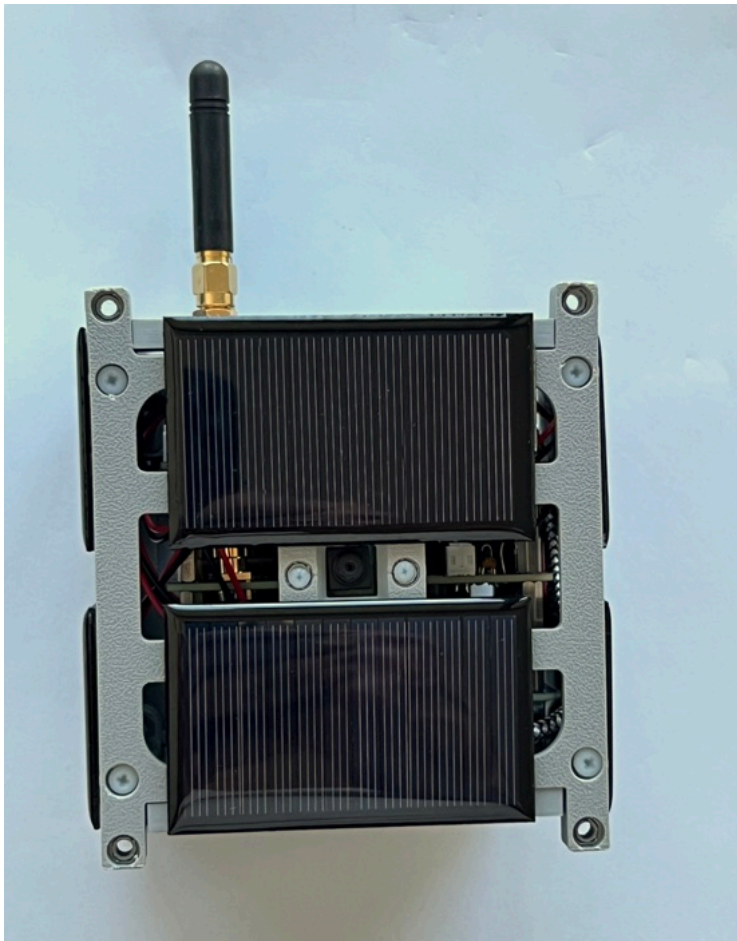
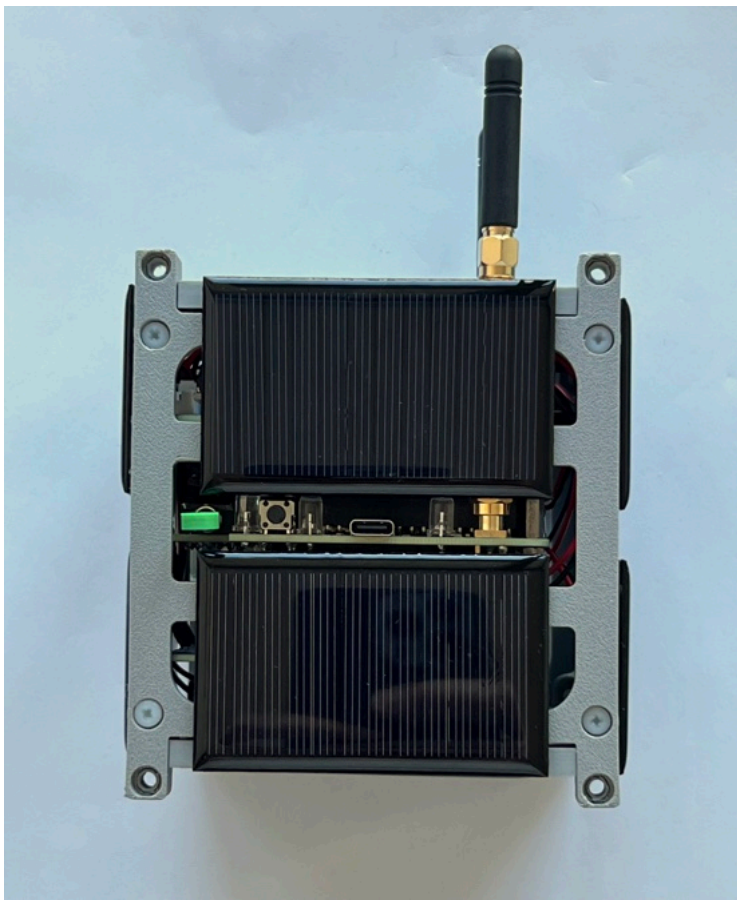
Next, screw the SMA antennas on to the coax:

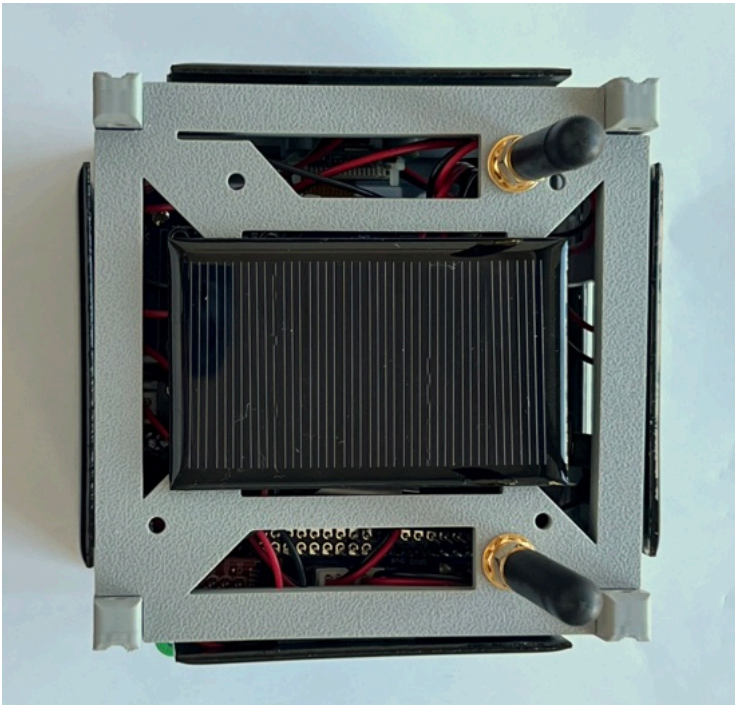
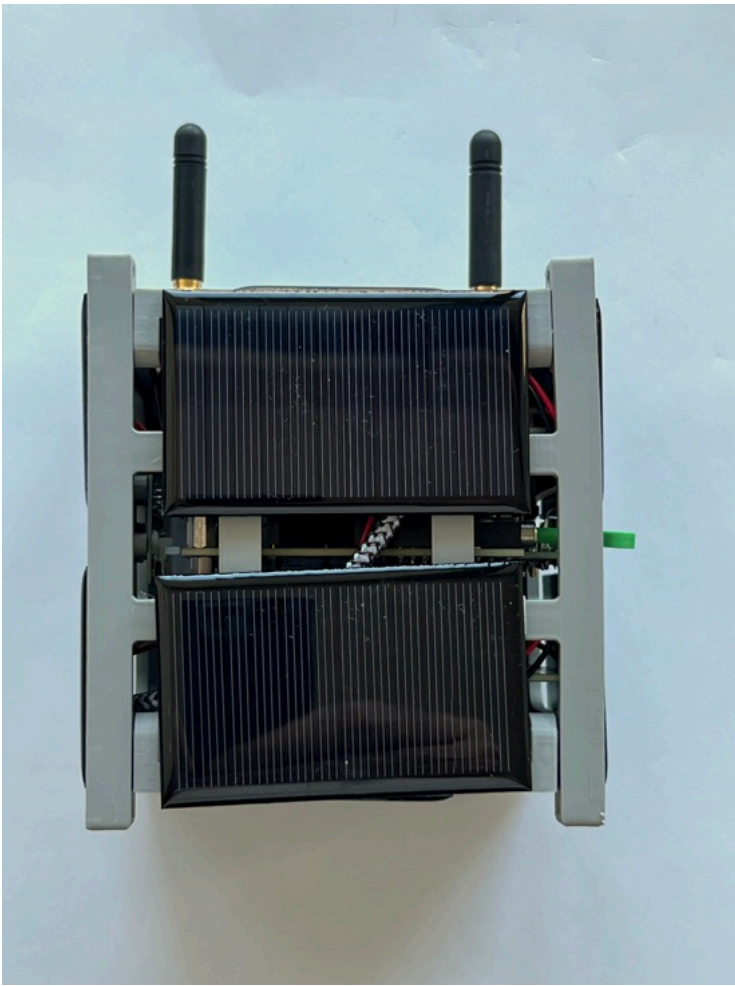


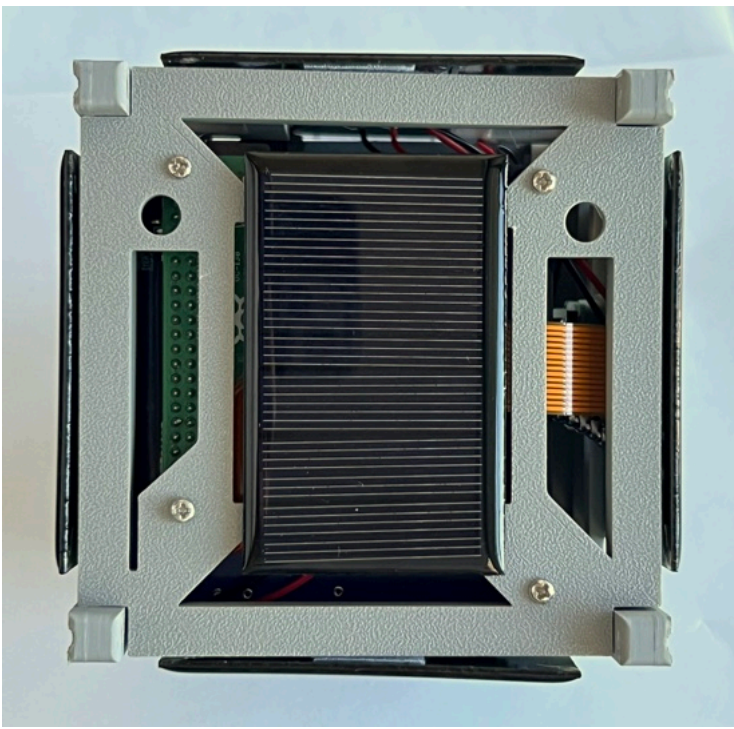


Here's how it looks:









Your CubeSatSim is complete and ready for [Step 9. Final Testing](#)

+ Add a custom footer

# 9. Final Testing

Edit New page

Alan Johnston edited this page 3 weeks ago · [1 revision](#)

## CubeSatSim Test Plan

### Using the CubeSatSim

After rebooting or powering up your Pi, tune your radio or SDR to 434.9 MHz (unless another frequency has been set) FM, and you should receive a signal. If you are testing with only the Pi (no CubeSatSim Main board plugged in), you will just hear the Morse Code (CW) of your callsign once (or the default AMSAT if you didn't supply one to the install script run the 'CubeSatSim/config -c' command). If you have build the Main board and plugged it into your Pi, you will hear telemetry signals as well after about 30 seconds.

On the Main Board, the green LED near the RBF switch and USB-C connector will be on when the CubeSatSim software is running. The red LED will illuminate when charging is occurring through the USB-C cable. The blue LED will illuminate when the CubeSatSim is transmitting.

- ▼ Pages 147
- Find a page...
- ▶ [Home](#)
- ▶ [1. Main Board 1](#)
- ▶ [2. Software Install](#)
- ▶ [3. Ground Station](#)
- ▶ [4. Main Board 2](#)
- ▶ [5. Battery Board](#)
- ▶ [6. Solar Board](#)
- ▶ [7. Solar Panels and Frame](#)
- ▶ [8. Board Stack](#)
- ▼ [9. Final Testing](#)
  - CubeSatSim Test Plan
  - Using the CubeSatSim
  - Quick Test
  - Command and Control Tests
  - Full Test
    - AFSK (APRS) Mode Tests
    - FSK/BPSK Mode Tests
    - FSK/BPSK Mode FoxTelem Tests



The pushbutton with the pi-power-button software will cause the Pi to reboot, change telemetry mode, shutdown, or change command and control mode. Pressing and holding the pushbutton will make the green power LED blink first once, then two times, then three times, then four times, then five times, then blink slowly, then blink even more slowly. Depending on when you release the button, different things will happen. Here's what happens if you:

- Press and release (don't hold button in at all): reboots CubeSatSim. The green LED will go out, and after 30 seconds, the CubeSatSim will be transmitting again.
- Press and release after one blink of green LED: switches to mode 1 which is APRS/AFSK telemetry mode. After about 30 seconds, the telemetry mode will switch to APRS.
- Press and release after two blinks of green LED: switches to mode 2 which is FSK/DUV mode. After about 30 seconds, the telemetry mode will switch to FSK.
- Press and release after three blinks of green LED: switches to mode 3 which is BPSK mode. After about 30 seconds, the telemetry mode will switch to BPSK.

Sensor Tests

SSTV Mode Tests

CW Mode Tests

Button Tests

Advanced Tests

Configuration Command Tests

Battery Discharge Tests

APRS or CW Telemetry Decoding Test

▶ [Adding New Sensors](#)

▶ [Command and Control](#)

▶ [Creating the CubeSatSim Raspberr...](#)

▶ [CubeSatSim Lite](#)

▶ [CubeSatSim Loaner User Guide](#)

Show 132 more pages...

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/alanbjohnston>



- Press and release after four blinks of green LED: switches to mode 4 which is SSTV mode. After about 30 seconds, a Scottie 2 SSTV image will be transmitted instead of telemetry. The first image transmitted is a stored image. If a Pi Camera is installed and enabled, a camera image will be transmitted every 30 seconds after that. Otherwise, a second stored image is repeatedly transmitted.
- Press and release after five blinks of green LED: switches to mode 5 which is CW mode. After about 5 seconds, the telemetry mode will switch to CW (Morse code).
- Press and release after green LED begins slow blinking: shuts down CubeSatSim.
- Press and release after green LED begins even slower blinking: changes the CubeSatSim command and control mode and reboots. If command and control via RF was ON, it will be turned OFF. If it was OFF, it will be turned ON after the reboot.

Once the CubeSatSim shuts down, the RBF pin can then be safely inserted. Removing the RBF pin will cause the CubeSatSim to start up again. It will use the same mode it was running when it was shutdown.

You can also change the telemetry mode using the command line. Typing `CubeSatSim/config` will show you the configuration options including how to change the mode. For example `CubeSatSim/config -s` will change to SSTV mode.

If you want to stop the CubeSatSim software, you can type these two commands:

```
sudo systemctl stop cubesatsim
```

```
sudo systemctl stop rpitx
```

After rebooting, the CubeSatSim software will start unless you disable it by typing:

```
sudo systemctl disable cubesatsim
```

You can then re-enable it on the next boot by typing:

```
sudo systemctl enable cubesatsim
```

## Quick Test

---

- Start with the CubeSatSim shut down and the RBF pin inserted.
- Remove the RBF pin to turn on the CubeSatSim. Within 30 seconds, the Green LED should come on, and a few seconds later the Blue LED should begin to blink.
- On power up, the CubeSatSim will automatically transmit on 434.9 MHz (unless a different transmit frequency has been set using the `CubeSatSim/config -F` command in the Raspberry Pi) when the Blue Transmit LED blinks. The first thing it will transmit is a Morse Code (known as CW for Continuous Wave) ID of HI HI DE CALLSIGN with FM modulation. (CALLSIGN will be the callsign you set by entering the `CubeSatSim/config -c` command in the Raspberry Pi. If you have never set it, it will be the default AMSAT) Listen with a radio or SDR on approximately 434.9 MHz FM demodulation to hear the Morse code callsign.

■ Depending on what mode the CubeSatSim is in, after the CW callsign, there will be delay of a few seconds or 30 seconds and then you should hear the telemetry or SSTV signal being transmitted. The Blue Transmit LED will sometimes be on, sometimes off depending on the mode.

■ Open FoxTelem in your Ground Station. If it isn't already stopped, click the `Stop` button. Go under the `File` menu then `Delete Payload Files` and click `OK`. The data in the `CubeSatSim-FSK` and `CubeSatSim-BPSK` tabs will now be all zeros. The `Telem Format` must be set to `BPSK 1200bps (Fox1E)` in the `Input` tab of FoxTelem and then click the `Start` button.

■ Set the CubeSatSim to BPSK mode (three blinks of the Green LED or by typing the `CubeSatSim/config -b` command in the Raspberry Pi) In the FFT window (red line), you should see the BPSK signal after the CW ID. Click on the center of the peak if it isn't already centered. Within a few seconds, the `Frames` count at the bottom of the window should increase by one every 4 seconds or so, and `Payloads` count at the bottom of the window should increase by three every 4 seconds..

■ Click on the `CubeSatSim-BPSK` tab then on the `Health` tab, and you should see some non-zero data and the `Telemetry Frames Decoded` count in the top right should be increasing every 4 seconds or so. In the box labeled `Battery`, look at the field for `Battery Voltage (V)`. This value should be in the range 3.5 V to 4.2 V. The `Battery Current (mA)` should be a few hundred milliAmps.



■ In the box labeled `Experiments`, it should say `STEM Payload Status OK` and there should be non-zero values in the fields below it. In the `+X Panel`, `+Y Panel`, or `+Z Panel` boxes, at least one of the `Acceleration` values should be non-zero.

■ Plug in the USB-C charger and make sure it is plugged into the wall and switched on. The red LED should illuminate. In the box labeled `Battery`, the `Battery Current (mA)` should be negative indicating the charging is happening.

■ Under the `Computer Hardware` box, if the `Camera` field says `OK`, then the Pi Camera is detected. To test SSTV and see the camera images, set the CubeSatSim to SSTV mode (four blinks of the Green LED or by typing the `CubeSatSim/config -s` command in the Raspberry Pi) and around 434.9 MHz FM. Run the SSTV decoding software. The first image should be the AMSAT logo with a white background. After that, images from the Camera should be transmitted. Camera images (not stored) display your callsign (or the default AMSAT) and the battery voltage and current (simulated battery voltage is not displayed) in black text.

## Command and Control Tests

---

■ To test the RF Command and Control features, you will need a radio which can transmit a FM modulated carrier in the amateur radio UHF band (default is 435 MHz but can be changed with the `CubeSatSim/config -F` command in the range 420 - 450 MHz). Always obey your local radio transmission rules and regulations. Command and Control can be used to change the telemetry mode instead of the pushbutton or Pi command or turn the beacon on or off.

■ Command and Control is turned off by default in the CubeSatSim software, so you will need enable it either by a Pi command or using the pushbutton. The Pi command is `CubeSatSim/config -T` and then type `y` for Yes. Using the pushbutton, press and hold the button in until the green LED blinks slowly then even more slowly. You can then release the pushbutton and the CubeSatSim will reboot and Command and Control mode will be enabled. You can verify the Command and Control state by the Pi command `CubeSatSim/config` and looking for the line `Radio carrier command and control is ON`

■ Command and Control can be done when the CubeSatSim is not transmitting, after the CubeSatSim has started up and transmitted the CW ID. You can tell this by looking for when the blue transmit LED is off or by listening on the receive frequency (434.9 MHz is the default, unless it has been changed). If you transmit for a few seconds as soon as the blue LED goes off, you should see the green LED blink a few times, and then the green LED and blue LED will go out, and the CubeSatSim will reboot. The number of times the green LED blinks tells you the new mode, which will be the next mode in sequence (the sequence is the same as when you press and hold the pushbutton: APRS->FSK->BPSK->SSTV->CW->APRS)

■ To turn off Command and Control mode, either use the Pi command `CubeSatSim/config -T` and then type `n` for No or using the pushbutton, press and hold the button in until the green LED blinks slowly then even more slowly and the CubeSatSim reboots.

■ For more Command and Control tests including using DTMF and APRS, see [Command and Control wiki page](#)

## Full Test

---

Complete the Quick Test above then these tests:

### AFSK (APRS) Mode Tests

■ Set the CubeSatSim to AFSK mode (one blink of Green LED or by typing the `CubeSatSim/config -a` command in the Raspberry Pi) and around 434.9 MHz FM (unless a different transmit frequency has been set using the `CubeSatSim/config -F` command in the Raspberry Pi), you should hear the CW ID and then within 30 seconds you should hear short APRS packets of about 2 seconds duration and see the Blue Transmit LED blink on and off with the packets.

■ A radio with an APRS TNC or a Ground Station with Direwolf or the OpenWebRX Web SDR set to DIG (Digital) Packet around 434.9 MHz will decode the APRS telemetry packets

■ One packet is sent about every 30 seconds.

■ The packets should contain: the CALLSIGN with a suffix of -11 and latitude and longitude coordinates (Direwolf will print the latitude and longitude numbers. OpenWebRX will map it if you click on the icon). The values should be the ones you set using the Raspberry Pi command `CubeSatSim/config -c` for the callsign and `CubeSatSim/config -l` (lower case L) for the latitude and longitude, or if you never set them, they will be the default callsign AMSAT and the default coordinates of Washington, DC, AMSAT's headquarters.

■ The APRS icon should be a satellite, unless Balloon mode has been set using the `CubeSatSim/config -B` command in which case the icon will be a balloon.



■ The telemetry string should be displayed in the Comments field and should begin "hi hi" and then the battery voltage and current followed by a string with the sensor data.

## FSK/BPSK Mode Tests

■ Set the CubeSatSim to FSK mode (two blinks of the Green LED or by typing the `CubeSatSim/config -f` command in the Raspberry Pi) and around 434.9 MHz (unless a different transmit frequency has been set using the `CubeSatSim/config -F` command in the Raspberry Pi) FM, you should hear the CW ID and then you should hear the telemetry signal, which sounds like a low rumbling sound. If you zoom into the waterfall with an SDR, you can see two peaks, which represents the two binary values being transmitted.

■ Set the CubeSatSim to BPSK mode (three blinks of the Green LED or by typing the `CubeSatSim/config -b` command in the Raspberry Pi) and around 434.9 MHz (unless a different transmit frequency has been set using the `CubeSatSim/config -F` command in the Raspberry Pi) FM, you should hear the CW ID and then you should hear the telemetry signal a few moments later, which sounds like a buzzing sound. If you zoom into the waterfall with an SDR, you can see a rounded peak. Note that to hear BPSK properly, you should demodulate with USB (Upper Side Band).

■ For both FSK and BPSK, after the startup sequence, the signal will be transmitted continuously. The Blue Transmit LED will mostly be on, but will blink off every 4 seconds when a new frame of telemetry data is being sent.

## FSK/BPSK Mode FoxTelem Tests

■ In FSK or BPSK mode, FoxTelem version 1.09 or later will decode the telemetry. Having FoxTelem control an RTL-SDR or FunCube dongle is the simplest, although you might have to display the FFT (Fast Fourier Transform) spectrum (red line towards the bottom of the window) and click on the signal peak to tune it. The FFT spectrum is at the bottom of the Input tab and is a red line. If it isn't displayed, select Decoder/Show FFT and the red FFT line will be displayed. You might also need to drag a divider bar up from the bottom if your screen resolution is low. You may need to enter the Center Frequency of 434840 in the kHz box before clicking `Start` (don't enter 434900). You can also use an SDR or radio to tune the signal (FM demodulation for FSK and USB for BPSK) with the audio fed into FoxTelem, although this can be trickier to get working, especially with BPSK.

■ On boot, you can see the CW ID, then the telemetry signal around 434.9 MHz. It is a good idea to have "Find Signal" enabled. If you see either "Scanning" or "Faded" in light grey letters in the FFT window, this means Find Signal is enabled. If not, go under the `File` menu then `Settings` and under the `Decoder Options` check the box `Find Signal` then click `Save`. If you have never set your latitude and longitude in FoxTelem, you may need to do it before you can save the settings.

■ To decode the telemetry in FoxTelem, you need to have the `CubeSatSim-FSK` and `CubeSatSim-BPSK` spacecraft added on FoxTelem version 1.09 or later. If they don't appear as a tab, you can add them by selecting `Spacecraft` then `Add` then clicking on the `CubeSat_Simulator_DUV.MASTER` file to add `CubeSatSim-FSK` or `CubeSat_Simulator_PSK.MASTER` file to add `CubeSatSim-BPSK`.

■ With the `CubeSatSim` in FSK mode (two blinks of Green LED or `CubeSatSim/config -f` Pi Zero command), telemetry is easily decoded just by centering the decode on the signal. The `Telem Format` must be set to `Fox-1 200 bps DUV` in the `Input` tab of FoxTelem and the `Start` button clicked. In the `Input` tab, the `Frames` and `Payloads` counts at the bottom of the window should increase by one about every 4 seconds.

■ If you click on the `CubeSatSim-FSK` tab then on the `Health` tab, you should see some non-zero data and the `Telemetry Frames Decoded` count in the top right should be increasing.

■ FoxTelem decodes three types of telemetry frames: `RT` (Real-Time, instantaneous sensor readings), `MIN` (Minimum values stored since the last reboot or reset), and `MAX` (Maximum values stored since the last reboot or reset). Values of `0000` indicates no data has been decoded for this type of frames. `MIN` and `MAX` frames are sent about every 5 frames, so it might take up to 40 seconds to have received all three types of frames in FSK mode. If one of the columns stays at 0 for all the values, it means that type of frame has not been decoded.

■ The battery voltage ( Battery Voltage (V) , which means the three NiMH cells in series) should read in the range of 3.5 V to 4.2 V. If it is 3.9 V or higher, it is fully charged. If it is 3.7 V or lower, it is low on charge.

■ The Temperature field under Computer Hardware should vary slightly with each frame decoded, indicating that new values are being received.

■ Under Radio , the TX Antenna and RX Antenna should show as Deployed , although they will briefly show as Stowed in the first frame sent after powering up or changing modes.

■ Under Computer Hardware , the I2C Bus fields show the status of the two I2C buses. The normal state for a CubeSatSim is I2C Bus 1 OK , and I2C Bus 3 OK .

■ The Camera field will show OK if a Camera is plugged in and working correctly.

■ Your CubeSatSim will normally be sending real telemetry, as indicated by the Simulated Telemetry OFF setting. It can be configured to generate and send simulated telemetry using the Raspberry Pi CubeSatSim/config -t command then typing y for yes. If I2C Bus 3 shows as FAIL , the CubeSatSim will automatically switch to simulated telemetry.

■ Under Computer Software , Normal Mode should be OK , unless the battery voltage is below 3.55 V when it will show FAIL to indicate Safe Mode.

■ Under Battery , the battery voltage will be in the range 3.5 V to 4.2 V.



- When running under batter power, the battery current will be in the range of 200 - 450 mA and will vary between samples.
- When charging with the USB-C charging cable or through the micro USB cable attached to the Pico, the current will be negative and the battery voltage greater than 4.0 V. When the battery is fully discharged, the battery charging current will be over 500 mA. When fully charged, the battery charging current will be less.
- When the USB-C charging cable is unplugged, the battery voltage will always decrease. When the USB charging cable is plugged in, the battery voltage will always increase. If the battery is charged, it will be close to 4 V.
- Under `Battery2` , the voltage and current will read zero unless you have a second battery board on your CubeSatSim.
- The `+X Panel` , `+Y Panel` , and `+Z Panel` boxes show telemetry of the solar panel voltage, current, and acceleration (gravity). The `-X Panel` , `-Y Panel` , and `-Z Panel` only show voltage and current.
- A voltage of 0 indicates the sensor is unavailable. Under no illumination or with no solar panel plugged in, the voltage will read around 0.8 V to 1 V.
- Under room illumination, you will see a voltage of around 2 V.
- Under lamp illumination or sunlight, you will see a voltage up to 5 V.

■ You will only see a non-zero current under lamp illumination or sunlight. LED lamp illumination will be less than 10 mA, while sunlight or halogen lamp illumination can be up to 200 mA. Halogen lamps put out lots of heat - don't leave the CubeSatSim in front of a halogen lamp for more than a few seconds.

■ Similar data can be displayed in FoxTelem when the CubeSatSim is in BPSK mode and BPSK 1200bps (Fox1E) is selected for the Telem Format . MIN and MAX payloads are sent every frame for BPSK, so they will update faster. The WOD tab will not show any data.

## Sensor Tests

■ In FoxTelem, under Experiments , if at least one of the BME280 or MPU6050 sensors is functioning, it will say STEM Payload Status OK . If neither are plugged in or functioning, it will show as STEM Payload Status FAIL . If it is OK, there will be non-zero data at least one of the two sensors. The rest of the tests in this section assume the status is OK.

■ If the CubeSatSim is at rest, the Rotation under the +X Panel , +Y Panel , and +Z Panel will be close to zero.

■ If the CubeSatSim is moved or bumped, the Rotation values will spike, perhaps as high as +/- 50 dps (degrees per second).

■ With the CubeSatSim on the turntable on the Low setting, the Rotation rate will be about 7 -9 degrees per second. If it is sitting upright (+Z Panel facing up), the Rotation value will be in the +Z Panel .

- Turn the CubeSatSim on its side on the turntable, and the rotation axis will change to the side that is facing up (e.g. +/-X Panel or +/-Y Panel )
- If it is tilted at an angle on the turntable, more than one Rotation axis will show a non-zero value.
- If it is rotating, the Green LED on the +Y side of the Pico board will illuminate, and be off if it is still.
- If the CubeSatSim is at rest, the Acceleration field in the +X Panel , +Y Panel , and +Z Panel will indicate which way gravity is acting on the CubeSatSim. If it is flat on one side, one value will be around 1 g while the others will be close to zero.
- If it is tilted, more than one axis Acceleration will show a value. If the CubeSatSim is jostled, the Blue LED on the +Y side of the Pico board will illuminate, and be off if it is not being sharply moved.

## SSTV Mode Tests

- If in SSTV mode (four blinks of the Green LED, or typing the CubeSatSim/config -s command in the Raspberry Pi), the sound of SSTV tones should start after the CW ID around 434.9 MHz FM (unless a different frequency has been set with the CubeSatSim/config -F command in the Raspberry Pi)
- The Scottie 2 SSTV image should be easily decoded using MMSSTV or QSSTV, with a short break between images.

■ If no Camera is plugged in or not working, the first image should be the AMSAT Logo on a white background after reboot or reset. Then, after that, every 30 seconds an image of the CubeSatSim will be transmitted on a black background as the Blue LED illuminates.

■ If a Camera is plugged into the Pi Zero, the first image after reboot or reset will be the AMSAT Logo on a white background. After that, every 30 seconds, the Camera will take a photo, then that photo will be transmitted as the Blue LED illuminates.

## CW Mode Tests

■ If in CW mode (five blinks of the Green LED, or typing the `CubeSatSim/config -m` command in the Raspberry Pi), you should hear Morse Code (CW) transmitted around 434.9 MHz FM (unless a different frequency has been set).

■ The CW will be HI HI then a sequence of numbers representing the telemetry.

## Button Tests

■ Pressing and holding the button until the Green Power LED blinks once will put the CubeSatSim into APRS mode after a reboot. If it is already in APRS mode, nothing will happen.

■ Pressing and holding the button until the Green Power LED blinks twice will put the CubeSatSim into FSK mode after a reboot. If it is already in FSK mode, nothing will happen.



■ Pressing and holding the button until the Green Power LED blinks three times will put the CubeSatSim into BPSK mode after a reboot. If it is already in BPSK mode, nothing will happen.

■ Pressing and holding the button until the Green Power LED blinks four times will put the CubeSatSim into SSTV mode after a reboot. If it is already in SSTV mode, nothing will happen.

■ Pressing and holding the button until the Green Power LED blinks five times will put the CubeSatSim into CW mode after a reboot. If it is already in CW mode, nothing will happen.

■ Pressing and releasing button will reboot, and CubeSatSim will start back up in 40 seconds in the same mode.

■ To power back up a CubeSatSim, remove the RBF pin. If it is not powered up and the RBF pin is removed, the battery might be drained. Plug in the USB-C charger or micro USB cable to Pico and it should power up.

## Advanced Tests

---

### Configuration Command Tests

Note: These tests are for custom configuration of your CubeSatSim by logging into the Raspberry Pi Zero.

■ Log into your Raspberry Pi Zero using a micro USB cable or over WiFi using ssh. See [Step 2 Software](#) for how to do this.

■ Verify your CubeSatSim configuration by typing `CubeSatSim/config` . The current settings and configurations will be listed. See all the possible settings and diagnostics by typing `CubeSatSim/config -h` . The output will list all the possible commands by letter (note some are upper case and others are lower case).

■ Verify you can change modes using commands. Change to FSK mode by typing `CubeSatSim/config -f` Then, after verifying that it is in FSK mode, change to BPSK mode by typing `CubeSatSim/config -b`

■ Change the callsign by typing:  
`CubeSatSim/config -c` then entering the callsign, such as W3YP, in all capitals then  
Return .

■ Change the latitude and longitude by typing:  
`CubeSatSim/config -l` then entering the numbers then Return . Change to AFSK mode to verify that it changed.

■ Test manually reading the current and voltage sensors by typing `CubeSatSim/telem`  
You should see seven readings.

■ Do a scan of the I2C buses by typing `CubeSatSim/config -S` (capital S). You should see multiple sensors listed at different addresses on each bus.

■ Turn on Simulated Telemetry by typing `CubeSatSim/config -t` then typing `y` for yes. In FoxTelem you will see `Simulated Mode ON` and also the values changing. Make sure you change it back by running `m` then typing `n` for no or you will be confused next time you run your CubeSatSim!

## Battery Discharge Tests

■ To test the battery discharge, keep running in BPSK mode for at least 10 minutes. In FoxTelem, you will see the battery voltage fall over time.

■ Plug in the USB-C charging cable. The Red Charging LED will illuminate and FoxTelem or v will show a negative battery current indicating charging.

## APRS or CW Telemetry Decoding Test

■ With the CubeSatSim in APRS or CW mode, copy the telemetry string beginning with "hi hi" followed by a string of 3 digit numbers.

■ Copy and pasting the telemetry string starting with "hi hi" into even rows of Data Input sheet of the telemetry spreadsheet (<https://cubesatsim.org/telem>) should show some currents, voltages, and temperatures. Note that only solar panel current is displayed, so only strong illumination will generate charging current.

+ Add a custom footer



# Adding New Sensors

Edit

New page

[Jump to bottom](#)

Alan Johnston edited this page last week · [3 revisions](#)

This page describes how to add new sensors to the Main Board.

First, you will need to wire your sensor to the Pico on the Main board. Most sensors are I2C, and the easiest way to add them is to use one with a Qwiic connector on it. The Main board has a Qwiic connector, so you will just need a Qwiic sensor and cable such as these:

<https://www.sparkfun.com/qwiic> Alternatively, you can wire it up manually.

Secondly, you will need to install the Arduino IDE software so you can compile the Payload code that runs on the Raspberry Pi Pico microcontroller.

## Blink Test

To do the blink test, you need to program your microcontroller. Here are the basic instructions for the Raspberry Pi Pico or Pico W.

I'd recommend using the Arduino Integrated Development Environment (IDE) application to program your Pico, although you can use other IDEs too. Use the 1.8.x version instead of the new version 2 since the new version has some issues with the Pico board. Download it here:

<https://www.arduino.cc/en/software> You will need to scroll down to find the "Legacy IDE (1.8.X)" version to download.



## Legacy IDE (1.8.X)



### Arduino IDE 1.8.19

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Arduino IDE 1.x documentation](#) for installation instructions.

#### SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

#### DOWNLOAD OPTIONS

**Windows** Win 7 and newer

**Windows** ZIP file

**Windows app** Win 8.1 or 10 

**Linux** 32 bits

**Linux** 64 bits

**Linux** ARM 32 bits

**Linux** ARM 64 bits

**Mac OS X** 10.10 or newer

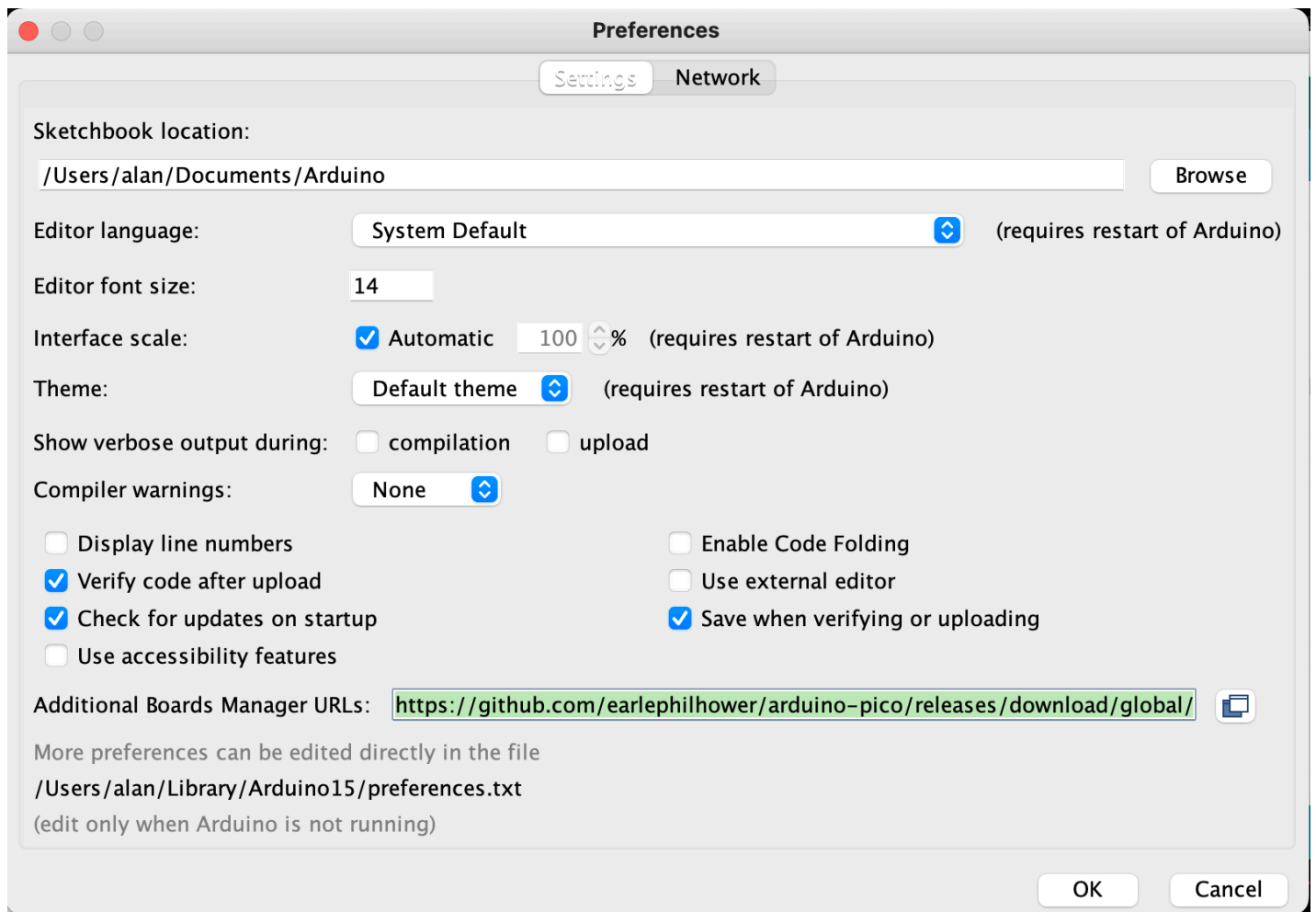
[Release Notes](#)

[Checksums \(sha512\)](#)

You can even install the Arduino IDE on your Raspberry Pi Ground Station using the "Linux 32 bits" version.

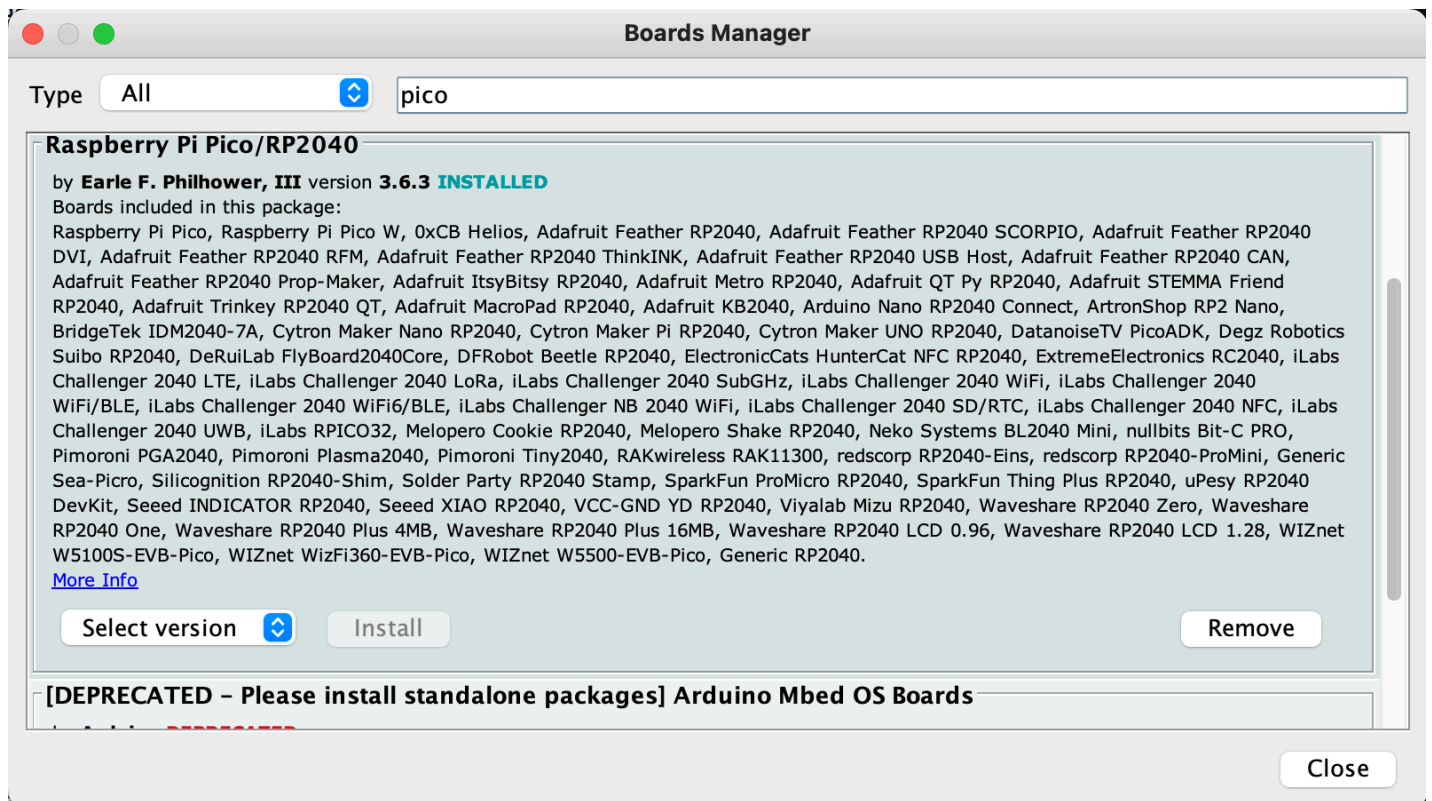
Before you can program your Pico using the Arduino IDE application, you will need to install the board files for the Pico. First, you need to go under Arduino then Settings to open the Preferences and look for the Additional Boards Manager URLs field. Type this URL

`https://github.com/earlephilhower/arduino-pico/releases/download/global/package_rp2040_index.json` into the Additional Boards Manager URLs field:



Select OK. Then open the menu Tools/Board/Boards Manager which will open the Boards Manager. In the search box, type `pico` then return. You will see the Raspberry Pi Pico/RP2040 by Earle F. Philhower, III Board file which supports the Raspberry Pi Pico and Pico W board.

Click Install and after a few minutes it will show as Installed.



Close the Boards Manager. Now open the Blink example code under the menu File/Examples/01.Basic/Blink and open this sketch.

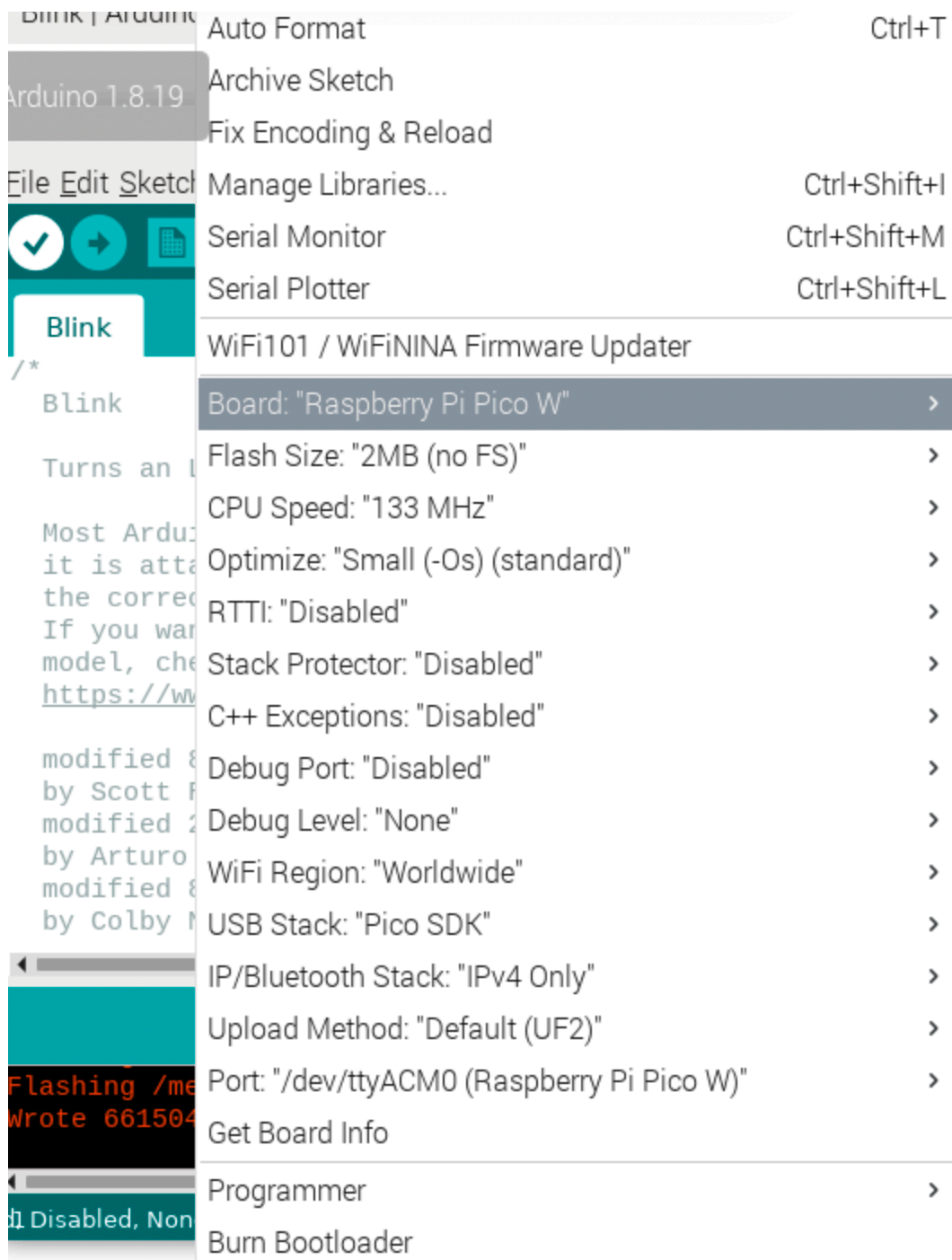
Go under the Tools menu and select Board and select Raspberry Pi RP2040 Boards. You also need to select the correct version of the board. Select the Raspberry Pi Pico W (this will work even if you have a Raspberry Pi Pico without WiFi). If your Pico has never been programmed, you won't need to select the Port. Instead, you will notice a USB flash drive called RPI-RP2 that has mounted on your computer.

Click the Verify icon (check symbol) at the top to Compile the code. If that works, click the Upload icon (arrow pointing right) to upload it to the Pico. You may get a popup about a drive being unplugged - just ignore this, it is normal. If all goes well, the built-in LED in your Pico should be blinking on and off!



Now if you look under Tools/Port you will see a new port labeled Raspberry Pi Pico W. You may have to select this port manually in the future after you plug in your Pico to reprogram it.





If the Upload fails with an error, you might need to manually select the Port. Under menu Tools/Port look for one that says Raspberry Pi Pico W and select it. If you can't find it, try unplugging the Pico, noting the ports listed, then plugging it back in. The new port listed will be the Pico.

If you have difficulty programming your Pico, see the notes here:

<https://github.com/earlephilhower/arduino-pico#installing-via-arduino-boards-manager>

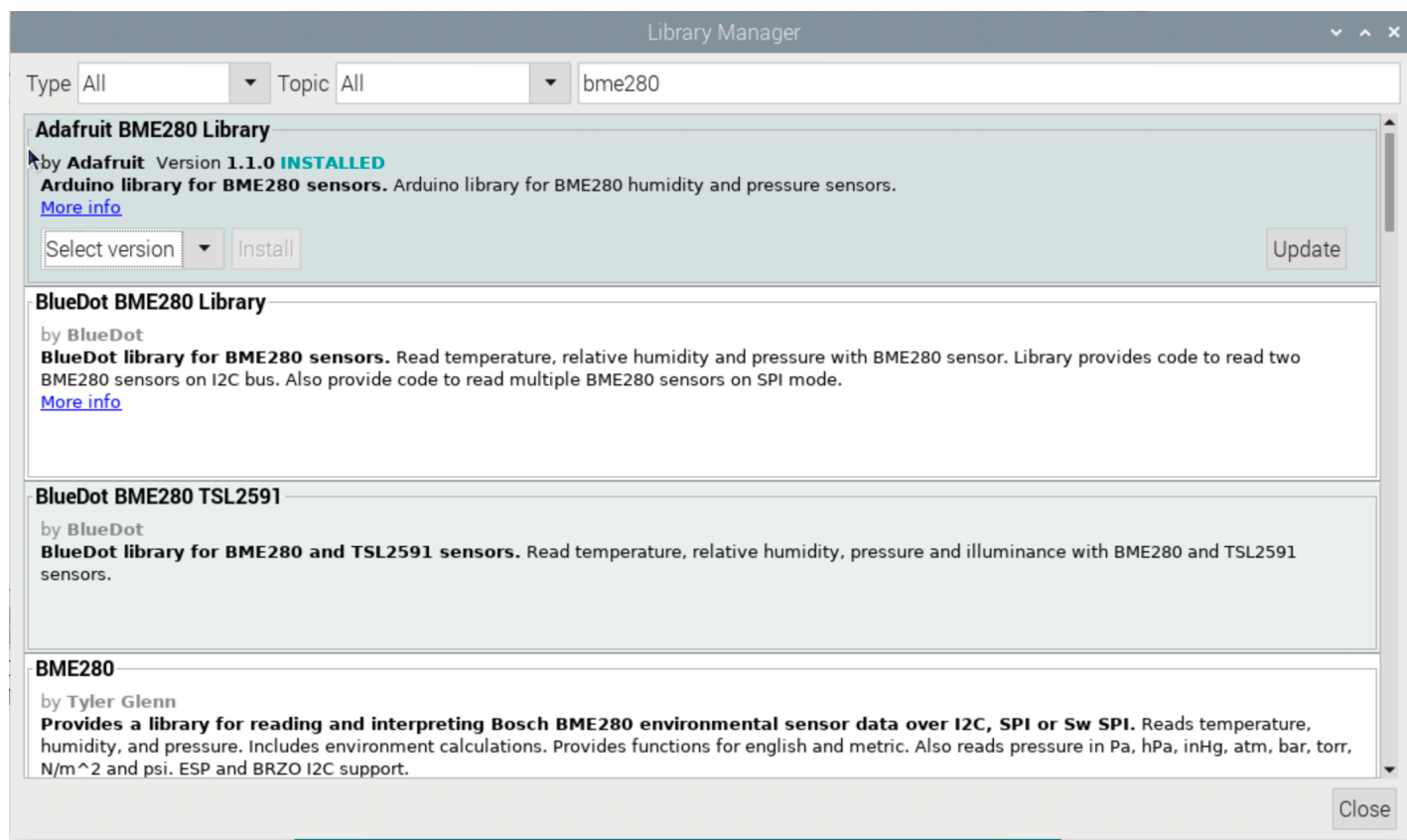
## Payload Sensor Testing

You can now test that the Pico can communicate with the two sensor boards. Here is the code (sketch) that runs on the Pico:

[https://github.com/alanbjohnston/CubeSatSim/blob/master/stempayload/Payload\\_BME280 MPU6050\\_XS/Payload\\_BME280 MPU6050\\_XS.ino](https://github.com/alanbjohnston/CubeSatSim/blob/master/stempayload/Payload_BME280 MPU6050_XS/Payload_BME280 MPU6050_XS.ino)

This code requires several libraries which you will need to install. If you forget to install one or more libraries, you will get error messages saying "No such file or directory" when you Verify or Upload the code.

Open Tools/Manage Libraries and the Library Manager will open after a few moments. In the search box, type `bme280` then return and you will see the Adafruit BME280 library. When you mouse over it, it will give a version pulldown menu and an Install button. Select version 1.1.0 and click on Install to install it. You will be prompted to install a few other library dependencies as well - select Install All. It should look like this when it completes:



Next, type `tockn` then return in the search box and install the MPU6050\_tockn library and click Install to install it. Type `tinygpsplus` into the search box and install this library as well.

Click Close to close the Library Manager. In the sketch window, click on the Verify icon (checkbox) and it should compile without errors.

```
Arduino 1.8.19
Payload_BME280_MPU6050_XS | Arduino 1.8.19
File Edit Sketch Tools Help
Verify
Payload_BME280_MPU6050_XS payload_extension.cpp
// code for Pico or Pro Micro or STM32 on the CubeSat Sim
// works wih CubeSatSim software v1.3.2 or later
// extra sensors can be added in payload_extension.cpp

#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>
#include <MPU6050_tockn.h>
#include <EEPROM.h>
#include <TinyGPS++.h>

#define SEALEVELPRESSURE_HPA (1013.25)

Adafruit_BME280 bme;
MPU6050 mpu6050(Wire);
TinyGPSPlus gps;

long timer = 0;
int bmePresent;
```

Done compiling.

Sketch uses 330500 bytes (15%) of program storage space.  
Global variables use 72352 bytes (27%) of dynamic memory,

d@disabled, None, Pico SDK, Worldwide, IPv4 Only, Default (UF2) on /dev/ttyACM0

If you get an error saying "No such file or directory", go back to the Library Manager and make sure you have installed the correct version of all three libraries. Click on the Upload button (right arrow) and the Pico should be programmed to read the sensors.

Once it is done uploading (indicated by the Done Uploading) message at the bottom of the window, open the Serial Monitor by clicking on the magnifier icon in the top right of the window. You should get a response displayed similar to this updated every second or so:

```
OK BME280 24.89 1003.96 77.61 21.13 MPU6050 -0.85 -2.48 -1.09 0.19 -0.15 1.02 GPS 0.0 0.0 0.0
TMP 22.3
```

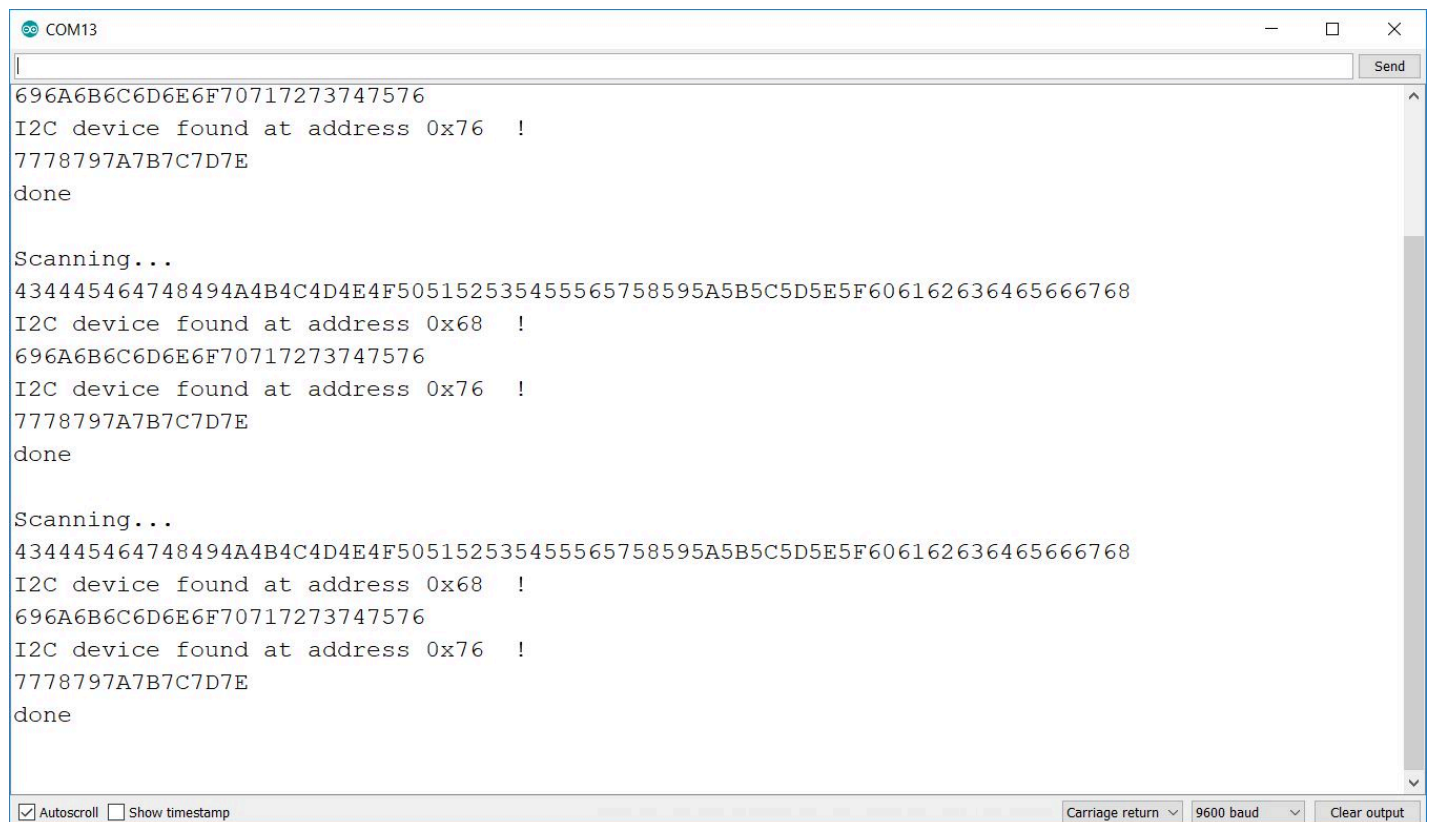
The `OK` is the status response. The four numbers after the `BME280` are the temperature in Celsius, pressure in hPa, altitude in meters, and humidity in percentage read from the purple BME280 sensor. The six numbers after the `MPU6050` are the X, Y, and Z axis angular rotation in degrees per second and the X, Y, and Z axis acceleration in g. If you get a series of zeros after a sensor, it means it was not successfully read by the Pico. The three numbers after `GPS` are GPS latitude, longitude, and altitude which will be zero unless you have connected a GPS module to your board. After the `TMP` is the temperature estimated from the diode D3.

On the Raspberry Pi, you can also use the `CubeSatSim/config -p` command to see the sensor data from the Pico.

If you get all zeros for a sensor, you need to determine if it is a hardware or software problem. Running an I2C bus scanner program will tell you if the sensor can be accessed on the I2C bus. The program is:

[https://github.com/alanbjohnston/CubeSatSim/tree/master/stempayload/i2c\\_scanner](https://github.com/alanbjohnston/CubeSatSim/tree/master/stempayload/i2c_scanner)

Upload this to your Pico and then open the Serial Monitor. You should see:



```
COM13
696A6B6C6D6E6F70717273747576
I2C device found at address 0x76 !
7778797A7B7C7D7E
done

Scanning...
434445464748494A4B4C4D4E4F505152535455565758595A5B5C5D5E5F606162636465666768
I2C device found at address 0x68 !
696A6B6C6D6E6F70717273747576
I2C device found at address 0x76 !
7778797A7B7C7D7E
done

Scanning...
434445464748494A4B4C4D4E4F505152535455565758595A5B5C5D5E5F606162636465666768
I2C device found at address 0x68 !
696A6B6C6D6E6F70717273747576
I2C device found at address 0x76 !
7778797A7B7C7D7E
done
```

The blue MPU6050 should be at address `0x68` while the purple BME280 should be at address `0x76`. If neither device is present on the I2C bus, make sure resistors R23 and R24 are soldered in and are 4.7k Ohms in value. If one sensor shows up but not the other, it might be due to soldering on the sensor pins or due to a bad sensor board.



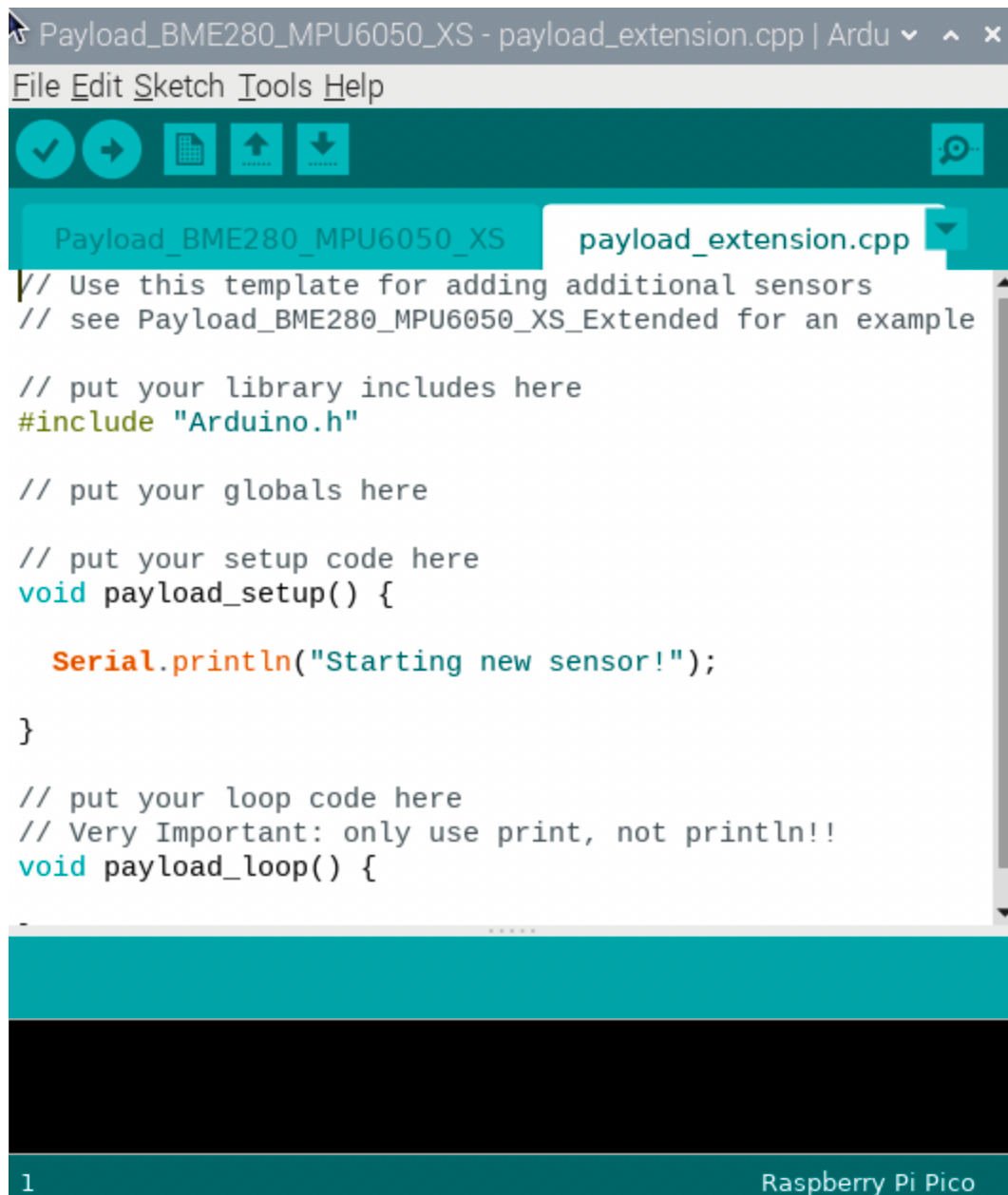
If both sensors show up on the I2C bus but you get zeros in the Serial Monitor, this indicates a software problem.

## Adding Sensors

---

You are now ready to start adding sensors to the code.

Click on the tab "payload\_extension.cpp" and you will see where you add the code for your sensor:



```
Payload_BME280_MPU6050_XS - payload_extension.cpp | Ardu v ^ x
File Edit Sketch Tools Help
[Icons: Checkmark, Arrow, File, Upload, Download, Search]
Payload_BME280_MPU6050_XS payload_extension.cpp
// Use this template for adding additional sensors
// see Payload_BME280_MPU6050_XS_Extended for an example

// put your library includes here
#include "Arduino.h"

// put your globals here

// put your setup code here
void payload_setup() {

    Serial.println("Starting new sensor!");

}

// put your loop code here
// Very Important: only use print, not println!!
void payload_loop() {

.....

1 Raspberry Pi Pico
```

There is an example in [Payload\\_BME280\\_MPU6050\\_XS\\_Extended](#) that you can use as a template.

Once you have added your code, simply compile and Upload to your Pico. You should see the new sensor at the end of the string in the Serial Monitor and in the APRS packet when it is decoded.

On the Raspberry Pi, you can also use the `CubeSatSim/config -p` command to see the sensor data from the Pico.

+ Add a custom footer

▼ Pages 147

Find a page...

▶ [Home](#)

▶ [1. Main Board 1](#)

▶ [2. Software Install](#)

▶ [3. Ground Station](#)

▶ [4. Main Board 2](#)

▶ [5. Battery Board](#)

▶ [6. Solar Board](#)

▶ [7. Solar Panels and Frame](#)

▶ [8. Board Stack](#)

▶ [9. Final Testing](#)

▼ [Adding New Sensors](#)

[Blink Test](#)

[Payload Sensor Testing](#)

[Adding Sensors](#)

▶ [Command and Control](#)

▶ [Creating the CubeSatSim Raspberry Pi Image](#)

▶ [CubeSatSim Lite](#)

▶ [CubeSatSim Loaner User Guide](#)

Show 132 more pages...

+ Add a custom sidebar

### Clone this wiki locally

<https://github.com/alanbjohnston/CubeSatSim.wiki.git>

